Multi-engine multi-level simulation for system specification validation and power consumption optimization

Fangyan Li

UNIVERSITE NICE SOPHIA ANTIPOLIS
POLYTECH'NICE-SOPHIA

École Doctorale des Sciences et Technologies de l'Information et de la Communication

Electronique pour Objets Connectés

THESE

Pour obtenir le titre de
Docteur en Sciences spécialité Electronique
de l'Université Nice Sophia Antipolis

présentée et soutenue par

Fangyan LI

Simulation multi-moteurs multi-niveaux pour la validation des spécifications système et optimisation de la consommation

Thèse dirigée par Gilles JACQUEMOD
Soutenance le 29 Mars 2016

Jury :

Y. DEVAL	Rapporteur	Professeur, IPB Bordeaux
I. O'CONNOR	Rapporteur	Professeur, Ecole Centrale Lyon
G. JACQUEMOD	Directeur	Professeur, UNS Sophia Antipolis
E. DEKNEUVEL	Encadrant	MCF, UNS Sophia Antipolis
R. BUTAUD	Examinateur	Ingénieur, Riviera Waves Sophia Antipolis
F. PECHEUX	Examinateur	Professeur, UPMC Paris

Avant-propos

Le travail présenté dans ce mémoire a été effectué au sein du Laboratoire EpOC, URE UNS 006, et cofinancé par la société Riviera Waves (convention Cifre). Il s'inscrit dans le cadre du projet CoCoE, soutenu par la Plateforme Conception CIM-PACA.

Je tenais à remercier …….

Table des Matières

Acronym

AA	Access Address
ADC	Analog to Digital Converter
AGC	Automatic Gain Control
AMS	Analog and Mixed Signal
AWGN	Additive White Gaussian Noise
BB and BBE	Base Band and Base Band Equivalent model
BER	Bit Error Rate
BT and BLE	BlueTooth and Bluetooth Low Energy
CoCoE	Contrôle de la Consommation Electrique
CRC	Cyclic Redundancy Check
DAC	Digital to Analog Converter
DSP	Digital Signal Processor (or Processing)
EDA	Electronic Design Automation
ELN	Electrical Linear Network
FIR	Finite Impluse Response filter
GF	Gaussian Filter
GFSK	Gaussian Frequency Shift Keying
IFT	Inverse Fourier Transform
IoT	Internet of Things
ISM	Industrial, Scientific and Medical band
LNA	Low Noise Amplifier
LPF	Low Pass Filter
LSF	Linear Signal Flow
LTF	Laplace Transfer Function
LL	Link Layer
MAC	Medium Access Control
MIM	Meet-In-the-Middle
MoC	Model of Computation
NF	Noise Figure
PDU	Protocol Data Unit
PSD	Power Spectrum Density
RMS	Root Mean Square
RTL	Register Transfer Level
SCAMS	SystemC-AMS (Analog Mixed Signal)
SCTLM	SystemC-TLM (Transaction Level Modelling)
SCNSL	SystemC Network Simulation Library
SNR	Signal to Noise Ratio
TDF	Timed Data Flow
TLM	Transaction Level Modelling

Introduction

1. Motivation

For several decades, the communicating objects invaded our everyday life and the number of transmitted data, associated with numerous standards of wireless communication, also exploded. Nowadays, objects are connected each other's, without sometimes human intervention, and connected to Internet; we speak then about Internet of Things (IOT). Many IOT applications require communicating wireless objects powered by autonomous battery for (mobility and easy to install) low cost and long autonomy (small reloadable battery, energy harvesting, low power consumption) systems. The transmission of a small quantity of data with regular interval of a large number of objects towards Internet imposes the use of a fixe (home, public places, etc.) or mobile (smartphone) gateway. The radio communication standards adapted to these applications are for example Bluetooth Low Energy (BLE) [1], ANT+ [2], or ZigBee [3].

In conclusion, the large-scale deployment of the internet of things generally and wireless sensors' networks in particular, requires the development of electronic circuits and systems more and more energy-efficient. Reducing the bill of materials and time-to-market can best be achieved through the broadest and most complete integration, one that also includes elements such as the antenna and/or the packaging where system-in-package (SiP) technologies are involved. It is clear that preparing them to handle the different signal shapes and frequencies is no trivial task. Indeed, during different design phases, engineers have to account for such factors as bit error rate (BER) on one hand, and power consumption on the other. Block design requires that electrical properties are set with due consideration for factors such as noise, nonlinearity, gain or impedance matching. Each step requires different simulators, and all must be interoperable. Even then, there is usually a design gap between the system specification and the RF block design. It typically centers on:

- the use of different design frameworks and different simulators;
- a huge frequency ratio between the RF (GHz) and the digital baseband (MHz).

Nevertheless, to meet an original system-level specification, engineers need to mix different levels of abstraction so that they can explore implementation architectures and then validate the final design at the circuit level. Luckily, existing behavioral models in the SystemC-AMS (or VHDL-AMS) analog and mixed signal language provide the basis for such a seamless system-to-transistor-level approach. Relevant features include:

- top-level functional simulations for architecture validation using a top-down methodology;

- bottom-up verification with accurately characterized models;
- test program development using tester resource models;
- traditional measurement, post-processing and/or the use of testbenches;
- IP exchange and protection to help assess a model against a specification.

This thesis proceeds from the position that designing such complex systems requires the development of a single EDA framework composed of multi-engine simulators that are associated with a library of hierarchical models. We have illustrated this approach in the domain of wireless remote sensors using BLE (Bluetooth Low Energy) standard for smart building application.

2. Context

This research belongs to the CoCoE project (**Co**ntrôle de la **Co**nsommation **E**lectrique dans les bâtiments) of ARCSIS (Pôle de compétitivité Solutions Communicantes Sécurisées) and CIM-PACA Design Platform. Partners of this project are EpOC (URE UNS 006) and IM2NP (UMR AMU-CNRS 7234) laboratories, Qualisteo and RivieraWaves companies. The objective of this project is to develop an innovative, non-intrusive and communicating solution for electrical energy measurement in the building. Today, optimizing energy in building is based on the total electrical power consumption, with no detailed information about power consumed in individual appliances. Thus, solution so far needs complex systems with many meter and sensors, which is incompatible and inconvenient with existing building.

The problematics, which must be solved by the CoCoE project, are that the electric power consumption information is less accessible, not in real-time without any information about kind of appliances and usage, even with the Linky future smart meter. This can be solved by using NIALM (Non-Intrusive Appliance Load Monitoring) technology implemented into FPGA and with the development of remote wireless sensors. There are two other PhD students working on this project in developing these new sensors to measure electrical current [4] and new NIALM algorithms with FPGA implementation [5] (Cifre Thesis with Qualisteo Company). The aim of this work, in collaboration with Riviera Waves Company (Cifre Thesis) is to model and optimize wireless communication systems (BlueTooth Low Energy, BLE standard) using SystemC-AMS language. This standard, BLE, will be used to interconnect the wireless current sensors in a building for appliance monitoring. The CoCoE project received an award during the World Efficiency Congress (Paris, October 2015): “Lauréat du Trophée de la Recherche Publique Energie-Climat-Environnement”, given by ADEME and the two magazines EnergiePlus and Mesures, in the topic of Energy efficiency. Moreover, CoCoE is a building block of the “Smart Campus Nice Sophia Antipolis” project, which has been labeled by the French ministry of industry on the national industrial plan on smart grid.

3. Thesis organization

The report of this thesis, besides an introduction and a general conclusion, includes three chapters:

Chapter I: Mixed signal modelling for BT communication. This chapter presents the description methodologies of the heterogeneous systems: top-down approach, Bottom-Up approach and meet-in-the-middle approach. Then, we remind the various RF front-end, as well as the different characteristics of RF segments and their modelling in Base Band (BB) equivalent circuit. We focus on the BT and BLE standard. To conclude this chapter, we show how the SystemC and SystemC-AMS languages can describe the different parts of a BLE transceiver.

- **Chapter II**: BLE transceiver modelling. In this chapter, we describe the different building blocks of a BLE transceiver. We remind the principle of the GFSK modulation to describe the block of bits generation. Thanks to the Riviera Waves Company design, a Zero-IF topology is chosen for the RF transmitter and a Low-IF for the RF receiver. Firstly, the "ideal" building blocks are described from a functional point of view and validated by simulations (integrator, correlator, modulator, filters, converter, … : in different domains: digital, analog and RF). Because of the prohibitive simulation time, the RF blocks are rewritten by using the BB methodology and then, refined to take into account the non-idealities which are going to impact on the couple consumption, BER. Every circuit (every model) is separately validated. Then a first system simulation (point to point communication between a transmitter and a receiver) is made. Finally the BER is estimated with suitable simulation time and the results were validated from measures on a circuit designed by the Riviera Waves Company.
- **Chapter III**: System modelling and simulation. This chapter is dedicated to the simulation of a real use case given by the Riviera Waves Company in order to verify the functioning of the system and to estimate the power consumption for a given use case. Several use cases are analyzed and compared. Finally, two versions of a same architecture are modelled, simulated and compared. The results show that the new version allows to increase the global performances of the system, in term of BER and power consumption. To conclude this chapter, we show that it was possible to reuse our models for a higher level simulation (sensors' network) by integrating them into the SCNSL platform, developed by the University of Verona to Italy, with whom we collaborated.

Chapter I – Mixed signal modelling for BT communication

1. Introduction

The design of wireless systems is becoming increasingly complex because of the great number of functional requirements for the application and non-functional constraints such as maximum energy consumption and minimum dependability. Simulation is strongly needed to verify that specifications are met and the simulation of communication aspects is crucial for the verification of today's wireless systems [6].

Figure 1.1 shows an example of a 2 nodes wireless communication system [7]. A complete wireless system like this can be divided into three different domains as: Digital Signal Processing (DSP), Analog BaseBand (BB) and Radio Frequency (RF) transceiver, according to the three different types of signal (mixed -digital and analog- and RF signal).

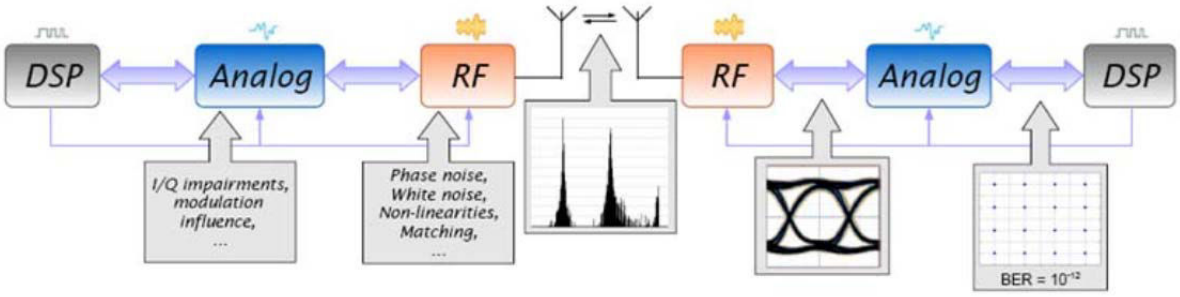


Fig. 1.1: Different domains of a 2 nodes wireless communication system and signals characteristics

A communication task involves all these parts but usually their behavior is simulated by using different tools and different models of computations. Hardware description languages (e.g., VHDL, Verilog, SystemC) are used for the whole architecture modelling and for the digital baseband (BB) block. Data flow simulators (e.g., Matlab/Simulink and SystemC-AMS) are used for optimization of signal processing algorithms, frequency planning, noise budget, bit error rate (BER) and power optimization. Specifically, there is a gap between system level and RF block simulation. RF circuit and communication channel handle radio waves and belong to continuous-time domain. The other components are digital and described by using discrete-time models of computation. Therefore, we need some techniques to handle heterogeneity.

Furthermore, the communication task consists of several aspects which cannot be simulated all together because of the huge different simulation time scale between domains. So we need to find out suitable compromises to be able to simulate different domains while keeping the accuracy of the specifications of interest.

In this chapter, we will first give a brief introduction about the problems of wireless mixed system modeling. In the next subsection, the methodologies focus on the problems solving and modeling considerations are talked, which involve: the system modeling approach and techniques choosing, introduction of different receiver types and the common RF circuit specifications considerations. The Bluetooth Low Energy standard will be presented in the third subsection with a focus on the Link Layer level and the Physical level. Briefly introductions of the common used tools and languages for mixed signal modeling and the chosen ones for this work are presented in the fourth subsection. And finally is a conclusion about this chapter.

2. Methodology

2.1. Mixed signal design methodology

Methodologies for the mixed design of wireless sensors network can be resumed as below, and illustrated by Figure 1.2. This approach uses a four-level hierarchical model library: transistor (SPICE or physical level), structural (low level), behavioral (intermediate) and high level (ESL or specification) [7]. It is also tailored to the three main types of design process: bottom-up, top-down and meet-in-the-middle [8]. The overarching goal is to define the primitive components and objects, and to then develop them and the system structure simultaneously, so that the final system is constructed from these primitives in the middle of the design process. The same paradigm is used to develop the hierarchical models.

The **Top-Down** approach is based on a design which begins with high level specifications. It is usually used for high level analysis and simulation based on logical and/ or behavioral models that take into account the system requirements.

This approach identifies the main objects of the system by analysis and successive refinements. The high level constraints can be associated to the objects (execution delay, consumption, packaging …). Thanks to the high level and logical models, the simulation time is greatly reduced which makes the architectural exploration easier.

The **Bottom-Up** approach can be viewed as an exploration of libraries containing models of physical solutions in order to build an architecture (virtual prototype) able to meet all the requirements, which guarantees the practicability of the system. But the complexity of the models increases the simulation time which is not convenient for architectural exploration.

The **Meet-in-the-Middle (MIM)** approach is based on the use of the lower level model outputs. It aims to extract more accurate performances from lower level design to enrich the higher level simulation. Generic parameters of high level models are extracted from low level simulation. The result of this operation is that we can obtain the accurate enough simulation results of system performances that we are interested in without executing lower level simulation. As the high level and low level are relative to each other, this method allows a flexible refinement or optimization between the lower levels and the higher levels.

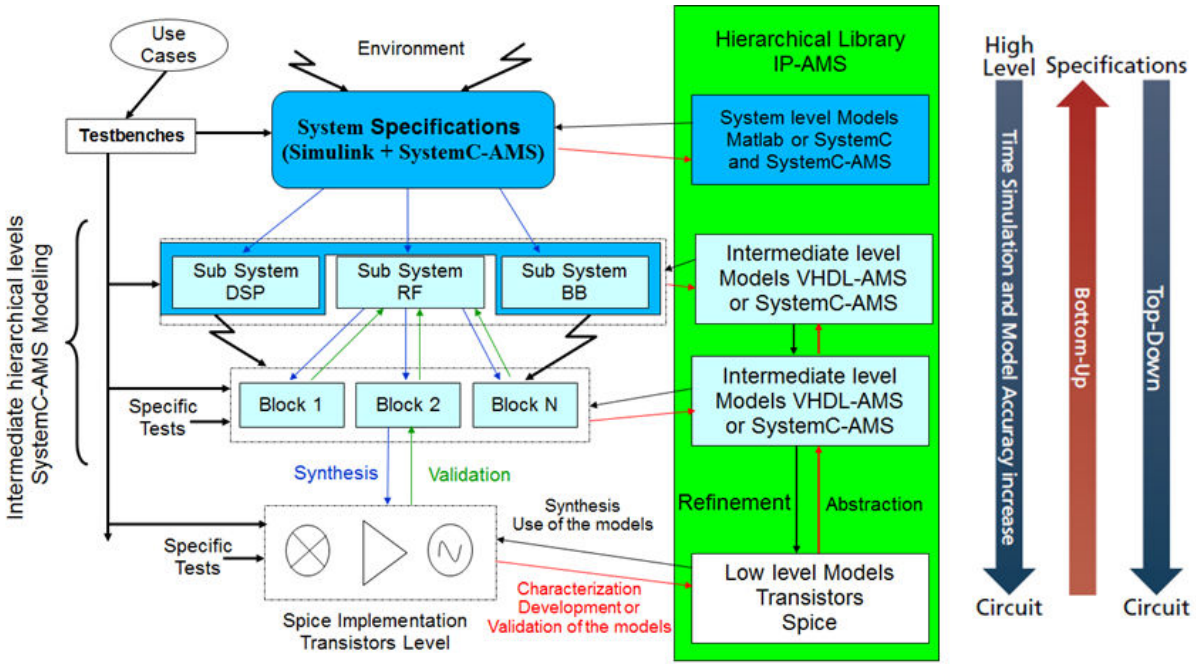


Fig. 1.2: Methodology for communicating object modelling

Example of the MIM approach

Figure 1.3 shows an example of a simplified open system interconnection (OSI) model which includes the highest "Upper levels" (higher than network level). Physical level is the lowest level in the OSI architecture. At the physical level, we find various building blocks, such as MODEM and analog/RF transceiver. At this level, the possible performances analyses are the transceiver Bit-Error-Rate (BER) and the power consumption. However, to be able to obtain a BER with the analog/RF component considerations, we need to extract the specifications of lower level analog circuits as the linear gain, noise figure (NF), non-linearity and the block average power.

In Fig. 1.3, the possible extracted performances of each abstraction level are written in a cloud symbol. They can be next used to interface with the higher level models to accelerate the global simulation. Several performances like the physical level BER can be used in all higher levels to perform as the error generator which is also a very important parameter for high level models. We can use the extracted specifications from different lower levels to

enrich different higher levels to realize the simulation we want. It reflects the flexibility of the MIM method.

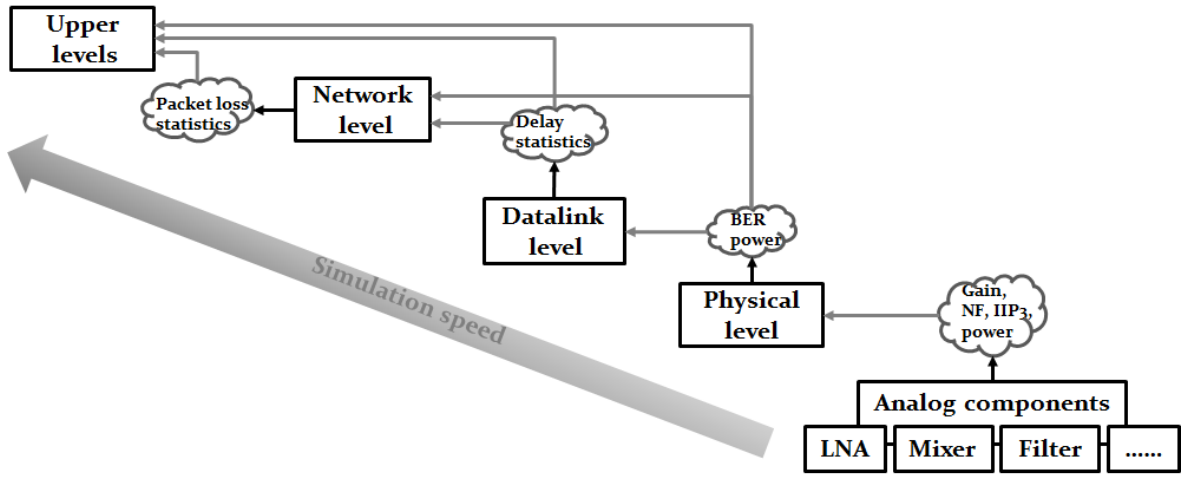


Fig. 1.3: Simplified OSI model with MIM method

The MIM method itself can not satisfy the physical level modeling due to the capability of modelling the RF blocks at system level. We traditionally over sample the signals by using a simulation frequency dozens of times of the highest signal frequency to keep the RF signal accuracy, which makes the simulation time very long and it is still not possible to co-simulate with Datalink and higher level system.

Limitations of the methodology

The main limitation concerns the development of the hierarchical library. Model development is difficult and needs a high level of expertise to be carried out effectively. Achieving an accurate RF implementation demands the full consideration of such factors as non-linearity effects, compression, noise, phase noise, frequency response, mismatch and so on. Moreover, RF designers also often use proprietary models during simulation, and this often means that there are limited opportunities for their reuse across different design environments.

Increased digital signal processing (DSP) lowers the quality of analog signal processing. Luckily, mixed-signal simulation is now a mature technology, and several unified simulators and co-simulation solutions are available [9]. This smoothes our path to the objective that, wherever possible, one should reuse existing models and unify various platforms in ways that are transparent to the user. The key factors for success are:

- the ability to take a hybrid approach to the integration of different tools;
- the use of open databases using standard languages;
- the availability of standard, compatible input/output formats and interfaces.

In this work, we use a combination of the MIM method and the Baseband Equivalent modeling method, which will be introduced in the next subsection, to realize a global system simulation with a suitable simulation time and it will give system performances with circuit level accuracy.

2.2. Baseband equivalent modelling

As we know that RF signal simulations, with the conventional oversampling method, require a very high oversampling rate to keep the RF signal accuracy. For example, a BLE RF signal is sent onto the 2.4GHz ISM band. To realize a simulation around 2.4GHz, the simulation frequency should attain 40 to 50 GHz, which makes the simulation out of the computer capacity or the tolerance in simulation time. In terms of narrow band communications like BT/BLE, since the useful in band information occupies very little part in the spectrum, this method waste most of the work on representing the out of band information.

The method of baseband equivalent modeling proposes to suppress the carrier frequency in order to accelerate the simulation speed without losing the RF/analog behavior. This method requires that the RF signal to be narrow band and the equivalent BB model should still be able to perform the RF non-idealities like the non-linearity. The principle of this method is briefly given below.

The principle of the modulation told us that we use the baseband information to vary the three elements, which are the amplitude, the frequency and the phase, of a sinusoidal wave, thus we can obtain the amplitude, frequency or phase modulated signal respectively. An RF modulated signal can be generally represented by the equation below regardless the modulation type we use.

$$s_{RF}(t) = Acos(\omega t + \int_0^t \theta(\tau)d\tau + \Delta\theta) \quad \text{(Eq.1)}$$

This equation can be written as

$$s_{RF}(t) = Re\{Ae^{j[\omega t+\varphi(t)]}\} \quad \text{(Eq.2)}$$

where $\varphi(t)$ represents the total instantaneous phase deviation $\int_0^t \theta(\tau)d\tau + \Delta\theta$. The carrier can be suppressed in the form of Eq.2 as:

$$s_{RF}(t) = Re\{Ae^{j\varphi(t)}\} \quad \text{(Eq.3)}$$

$$s_{BB}(t) = Ae^{j\varphi(t)} \quad \text{(Eq.4)}$$

The effect of the baseband equivalent modeling method in frequency domain is shown in Figure 1.4.

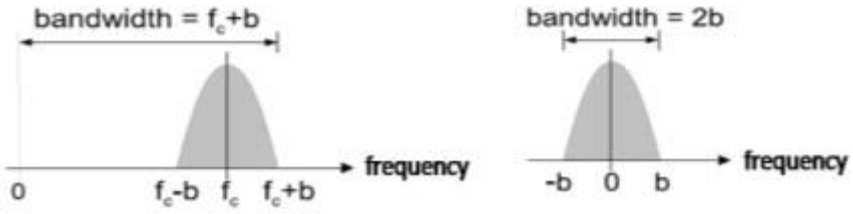


Fig. 1.4: Baseband equivalent modeling approach

In this case, baseband equivalent signal $s_{BB}(t)$ has to be modeled as a complex signal with two quadrature scalar components $\mathrm{I}(t)$ and $\mathrm{Q}(t)$ instead of one real signal as $s_{RF}(t)$.

Once these principles were put, we remind, during this chapter, the receivers' main architectures RF as well as the characteristics (parameters and equations).

2.3. Analog/RF front-end architectures

Architectures of a receiver can be divided into the digital and analog/RF parts and interfaced through an analog-to-digital converter (ADC). The analog/RF part in receiver is used to detect and receive the RF signal and convert it to baseband. It should be able to combat the interference and to have enough dynamic range. In this subsection, the different types of RF front-ends are introduced [10].

2.3.1. Heterodyne receiver

The architecture of heterodyne receiver, depicted in Figure 1.5, uses an intermediate frequency (IF) before to download the signal in the baseband spectrum.

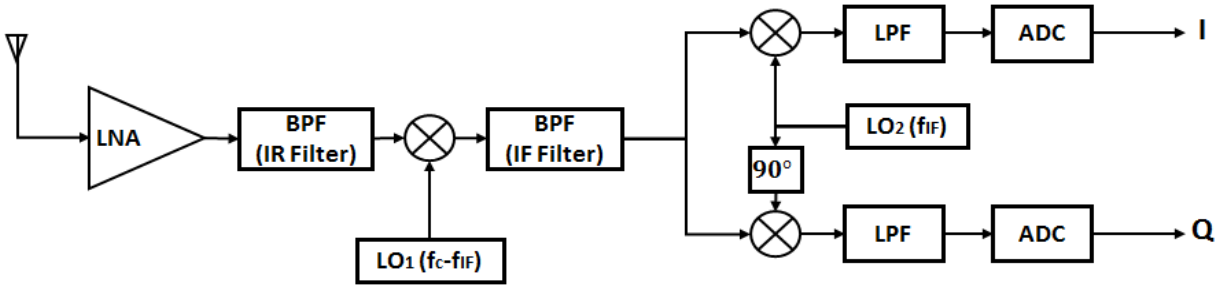


Fig.1.5: Heterodyne receiver block diagram

After the LNA, the RF signal is filtered by an image rejection (IR) filter to attenuate the image frequency, and then it is down converted to a first intermediate frequency (IF) IF_1 by

mixing an LO of $f_c \pm f_{IF1}$ where f_c is the RF signal carrier frequency or the channel frequency. This IF signal is next filtered by a high selectivity band-pass filter and then down-converted to the baseband by using a quadrature mixer. An ADC is used lastly for the signal digitizing. In comparing with the zero-IF, this architecture avoids the problems like LO leakage and flicker noise but with tradeoff of the complexity. Since it contains complicate filters and two mixing stages, it makes the design expensive and difficult to be integrated.

2.3.2. Homodyne (or zero-IF) receiver

Homodyne receiver, also called zero-IF receiver, has an architecture shown in Fig. 1.6. As other type of receivers, at the input of reception, we use a low-noise amplifier as the first reception block to pull down the chain noise figure (NF). Then it directly convert the desired RF signal down to the baseband with an LO which has the same RF frequency than signal carrier frequency. As the architecture is simple, this kind of receiver is easy to design and to be integrated. However, it suffers from the LO leakage, even-order distortion and the flicker noise.

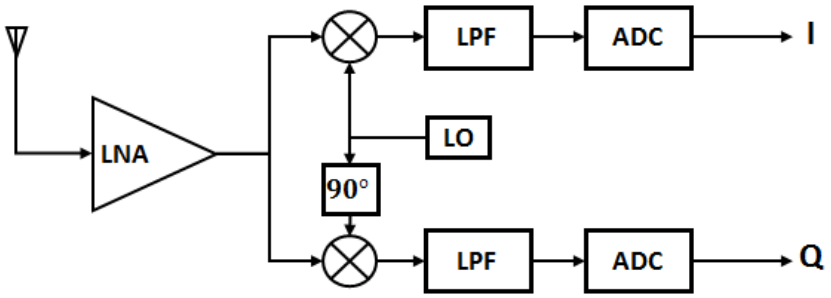


Fig. 1.6: Direct-conversion receiver block diagram

2.3.3. Low-IF (or pseudo zero-IF) receiver

The low-IF receiver, which has an architecture shown in the Fig.1.7, aims to avoid these problems by using an LO frequency shift a little bit from the carrier frequency of the received RF signal. The disadvantage of this architecture is that the ADC dynamic range has to be larger than in the zero-IF receiver.

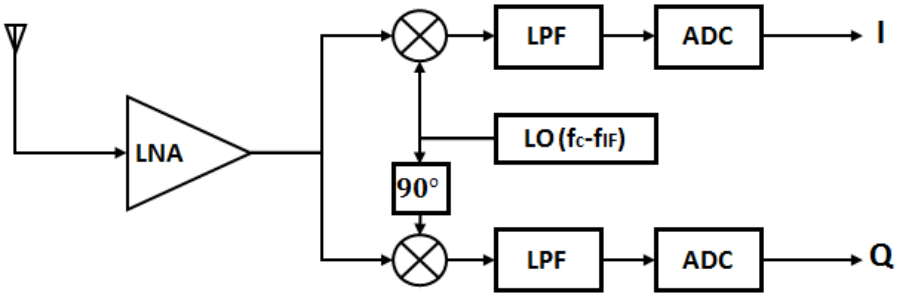


Fig. 1.7: Low-IF receiver block diagram

2.4. Analog/RF blocks modelling considerations using the MIM approach

To construct system-level models for RF circuits, we need to extract the useful information from low-level RF models or circuits and ignore other low-level details. In this part, we will talk about the mainly considered parameters in RF designs. It is necessary to understand the principle of them to use them lately in the system-level model refinements.

2.4.1. Noise modelling

Thermal noise

An additive noise n(t) is often introduced in communication system simulation to analyze the transceiver performances, as shown in Figure 1.8. We can use the PSD (Power Spectrum Density in W/Hz) for the analysis of Eb/N0 or the power (W) for the SNR studying. This noise called additive white Gaussian noise (AWGN) has a normal distribution in the time domain with an average value of zero.

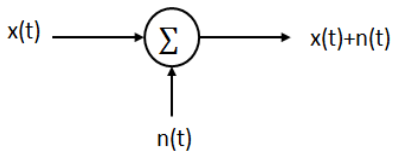


Fig. 1.8: AWGN model

AWGN is normally used to describe noises having Gaussian distribution like thermal noise. But in this work, we use an AWGN source to represent the general noise of an analog RF circuit. This noise is an abstraction of all kinds of the circuit internal noise and it is a method to measure the noisy level of a circuit. In the modeling stage, this value can be used to model the circuit noise as a thermal noise of the block equivalent input impedance to realize the block noise refinement.

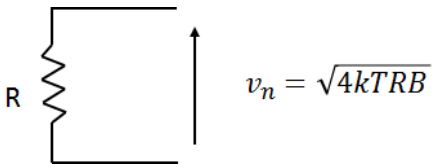


Fig. 1.9: Thermal noise of a resistance in tension

The thermal noise is dependent on the temperature, as shown below. For example, in Figure 1.9, a single passive resistor R generates an RMS noise voltage equal to

$$v_n = \sqrt{4kTRB} \qquad \text{(Eq.5)}$$

where k=1.38×10−23 J/K is Boltzmann's constant, T is the temperature in Kelvin, R is the value of the resistor in Ohms, and B is the bandwidth in Hertz.

Noise Figure and Sensitivity

The noise figure (NF) is an important analog/RF specification. It is used to quantify the impact of the thermal noise in communication systems. NF is defined as the noise factor (F) in decibels, and F is defined as the ratio between the input SNR_{in} (Signal to Noise Ratio) to the output SNR_{out} of an RF block:

$$F = \frac{SNR_{in}}{SNR_{out}} \quad \text{(Eq.6)}$$

The noise factor, F, of an RF block is always greater than one (zero in decibels) because there is always intrinsic noise in the block and the SNR of this block will be reduced. If we are up to a view of an RF system which is composed of cascaded blocks (cf. Figure 1.10), a receiver front-end composed by a low noise amplifier (LNA), a mixer, an analog filter and an ADC, the total noise factor can be obtained by using the Friis equation:

$$F_{total} = F_1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1 G_2} + \cdots + + \frac{F_n - 1}{G_1 G_2 \ldots G_{n-1}} \quad \text{(Eq.7)}$$

where F_n and G_n are the noise figure and the gain of the n_{th} stage respectively

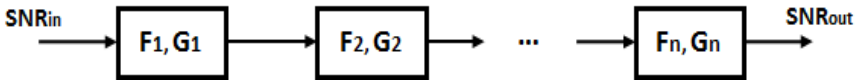


Fig. 1.10: Cascaded noisy blocks

From Eq.7, we can understand that the first stage of a receiver requires a very low noise figure and a high gain in order to obtain a good noise factor of a cascaded system.

The sensitivity is the minimum signal input level required for a specified output SNR of a receiver. The sensitivity of a receiver can be obtained from the required minimum SNR_o over the bandwidth B:

$\mathrm{SNR_{min}} = \frac{sensitivity}{\mathrm{F_{total}} \times KTB}$ and the sensitivity with an integrated thermal noise is obtained as:

$$\mathrm{Sensitivity} = \mathrm{F_{total}KTB} \times \mathrm{SNR_o} \quad \text{(Eq.8)}$$

It is expressed in dBm with the assumption of the temperature of 290 K:

$$\mathrm{Sensitivity_{dBm}} = -174 + 10\log \mathrm{B} + 10\lg \mathrm{F_{total}} + 10\log \mathrm{SNR_{min}} \quad \text{(Eq.9)}$$

AWGN in Simulation

Normally, AWGN can be generated by using a function with the RMS value equals to one. The function gives a series of pseudorandom numbers. In our simulation, AWGN is used to represent a white noise in the whole simulation band which is specified by the simulation sampling frequency fs. This noise should adapt the required noise quantity in the band of interest B.

The AWGN in a simulation is evenly distributed over all the simulation band $\pm f_s/2$. If we consider the noise in voltage, the RMS value of this noise v_n integrated in the band of interest B can be calculated with the defined block noise figure. For the simulation with a simulation frequency of f_s, we have to generate an RMS value equal to:

$$v_{n-simu} = V_n\sqrt{f_s} = v_n\sqrt{f_s}/\sqrt{B} \quad \text{(Eq.10)}$$

where V_n is the real RMS noise voltage density in $V/\sqrt{Hz}$, B is the system bandwidth, and f_s is the simulation sampling frequency (cf. Figure 1.11).

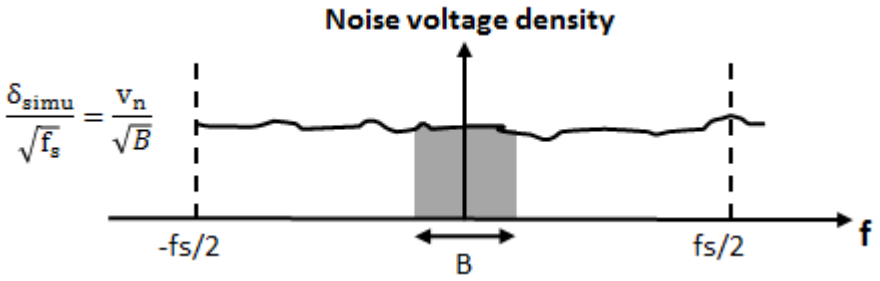


Fig. 1.11: Noise voltage density in simulation

In the classical case of thermal noise power $P_n = KTB$ that one encounters in many references for simulating the performance of a matched receiver as a function of the SNR or E_b/N_0, we have to generate an AWGN with a power equal to

$$P_{n-simu} = \frac{v_{n-simu}^2}{R} = \frac{KTBf_s}{B} = N_0 f_s \quad \text{(Eq.11)}$$

which gives an RMS noise voltage at receiver input of

$$v_{n-simu} = \sqrt{RN_0 f_s} \quad \text{(Eq.12)}$$

where R is the receiver input impedance.

2.4.2. Phase noise

In RF transceivers the complex baseband signal $x(t)$ containing the useful information will be up-converted in the transmitter or down-converted in the receiver by using RF mixers and LO output carriers. If the carriers are ideal, the mixer output signal $y(t)$ can be written with a complex $x(t)$ as::

$$y(t) = x(t)e^{j2\pi f_c t} \quad \text{(Eq.13)}$$

This mixer output can be written as below with the LO phase noise θ(t):

$$y(t) = x(t)e^{j(2\pi f_c t + \theta(t))} = x(t)e^{j2\pi f_c t}e^{\theta(t)} \quad \text{(Eq.14)}$$

The phase noise θ(t) is normally described by its PSD in dBc/Hz in the frequency domain. It is defined by the ratio between the noise power measured in 1 Hz bandwidth at a frequency offset f_Δ and the power of the carrier. The ideal oscillator is described by a sinusoidal output (in time domain) and is represented by a Dirac function in the frequency domain. Noises, such as flicker and thermal noise, generate a phase noise (in dBc/Hz@f_Δ) in the spectrum of a real oscillator output as described in Figure 1.12.

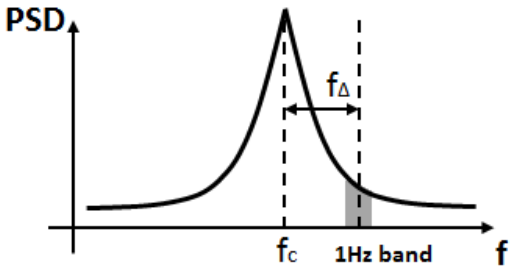


Fig. 1.12: Oscillator phase noise

PLL Phase Noise Modeling in time domain

In analog/RF design the PLL performances are generally defined by their output phase noise profile. PLL phase noise can be well specified by system designers using the model depicted in Figure 1.13.

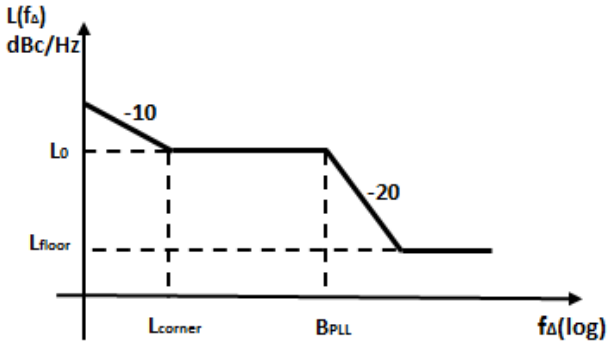


Fig. 1.13: Phase noise profile in dBc/Hz

In this work, we model a complete PLL phase noise in the frequency domain by using the method presented in [6] as :

$$L(f_\Delta) = \frac{B_{PLL}^2 L_0}{B_{PLL}^2 + f_\Delta^2}\left(1 + \frac{f_{corner}}{f_\Delta}\right) + L_{floor} \qquad \text{(Eq.15)}$$

in which f_Δ is the frequency offset from the carrier, B_{PLL} is the PLL −3dB bandwidth, and L_0 is the in-band phase noise level in rad^2/Hz.

This phase noise model can be seen as a low-pass filter for the BB modeling or a band-pass filter for RF modeling by shifting the zero value of f_Δ to f_c. We just need to convert this

model to a phase error source $\theta(t)$ by using the Inverse Fourier Transform (IFT), then add it to the LO carriers, as shown in Figure 1.14.

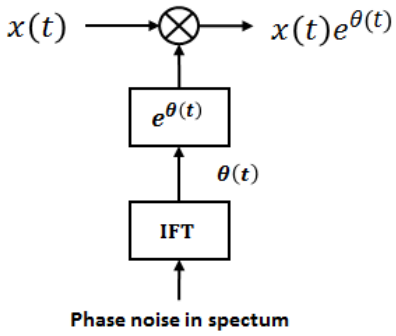


Fig. 1.14: Phase noise temporal simulation model

2.4.3. Non-linearity

When an analog/RF block is nonlinear, its input and output have a relationship approximated by

$$y(t) \approx \alpha_1 x(t) + \alpha_2 x^2(t) + \alpha_3 x^3(t) + \cdots \quad \text{(Eq.16)}$$

where $x(t)$ is the input and $y(t)$is the output. If the input is a single tone of sinusoidal signal as $x(t) = \text{A}\cos\omega\text{t}$, it will lead an output as

$$y(t) = \alpha_1 \text{A}\cos\omega\text{t} + \alpha_2 \text{A}^2 \cos^2 \omega\text{t} + \alpha_3 \text{A}^3 \cos^3 \omega t + \cdots$$

$$= \frac{\alpha_2 A^2}{2} + \left(\alpha_1 \text{A} + \frac{3\alpha_3 A^3}{4}\right)\cos\omega\text{t} + \frac{\alpha_2 A^2}{2}\cos 2\omega\text{t} + \frac{\alpha_3 A^3}{4}\cos 3\omega\text{t} \quad \text{(Eq.17)}$$

Where the first term is a dc offset, the second term is the fundamental wave, and the other high order signals are called nth-order harmonics which are integer multiples of the input frequency [11].

This result shows that a nonlinear circuit can generate other frequency components in the output with a single frequency input. We will next talk about several phenomena due to this property and the corresponding RF design specifications.

Gain compression

From Eq.17 we can see that the gain of the desired frequency is

$$G = \alpha_1 + \frac{3\alpha_3 A^2}{4} \quad \text{(Eq.18)}$$

and when the input amplitude A is very small, the other terms can be ignored since they all contains high order of A, then the output can be approximated as the input multiplied with the fundamental linear factor α_1. But as A becomes larger, the term $3\alpha_3 A^2/4$ becomes important and the combination with α_1 yields a nonlinear gain as shown in Figure 1.15. According to

different systems, the factor symbol of the different harmonics could be opposite. Fig. 1.15 (a) and (b) give respectively the development of the output gain of the fundamental signal according to the input amplitude with inphase α_1 and α_3 and inverted α_1 and α_3. Fig.1.15 (c) is the zoom of the common zone of them which indicates that the output can be seen as linear when A is small, but when A keeps arising, the output gain will be compressed.

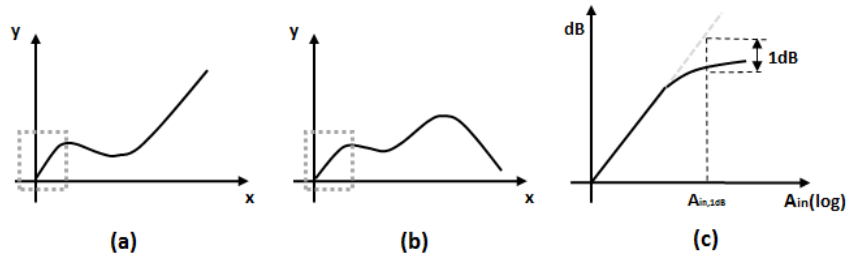


Fig. 1.15: Gain compression with: (a) inphase α_1 and α_3, (b) inverted α_1 and α_3, and (c) definition of 1-dB compression point

In RF design, this effect is measured by the specification of "-1dB compression point" (1dB CP) which defines the input amplitude $A_{in,1dB}$ which corresponds to a nonlinear output point of 1dB less than its ideal linear output as shown in Fig.1.15 (c) and Eq.19.

$$20\log\left|\alpha_1 + \frac{3}{4}\alpha_3 A_{in,1dB}^2\right| = 20\log|\alpha_1| - 1dB,$$

$$A_{in,1dB} = \sqrt{0.145\left|\frac{\alpha_1}{\alpha_3}\right|} \qquad \text{(Eq.19)}$$

In the situation of two tones at the nonlinear block input as $x(t) = \mathrm{A_1 cos\omega_1 t} + \mathrm{A_2} cos\omega_2 \mathrm{t}$, the fundamental output signal becomes

$$y(t) = \left(\alpha_1 + \frac{3}{4}\alpha_3 \mathrm{A_1^2} + \frac{3}{2}\alpha_3 \mathrm{A_2^2}\right) A_1 cos\omega_1 t + \cdots \qquad \text{(Eq.20)}$$

Cross-modulation

If $\mathrm{A_1} cos\omega_1 \mathrm{t}$ is the desired signal and $\mathrm{A_2} cos\omega_2 \mathrm{t}$ is an interferer, and $\mathrm{A_2}$ is much strong than $\mathrm{A_1}$, then the gain of the desired signal in Eq.20 is approximated by $\alpha_1 + \frac{3\alpha_3 A^2}{4}$. In the case of α_1 and α_3 are with reversed phase, the gain will be reduced rapidly until zero which is described as the desired signal is blocked and the interferer is called a blocker (cf. Fig. 1.16).

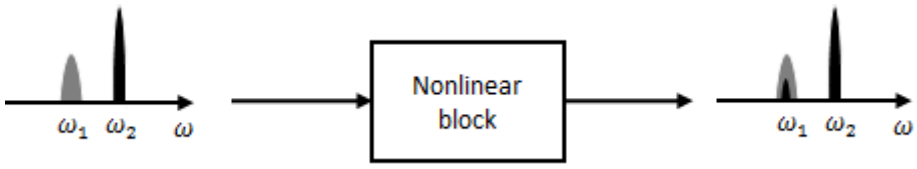


Fig. 1.16: Cross modulation

Intermodulation

Consider still two tones at the input as $x(t) = \mathrm{A}_1\cos\omega_1 \mathrm{t} + \mathrm{A}_2 cos\omega_2 \mathrm{t}$, the nonlinear output becomes:

$$\begin{aligned} y(t) &= \alpha_1(\mathrm{A}_1\cos\omega_1 \mathrm{t} + \mathrm{A}_2 cos\omega_2 \mathrm{t}) + \alpha_2(\mathrm{A}_1\cos\omega_1 \mathrm{t} + \mathrm{A}_2 cos\omega_2 \mathrm{t})^2 + \cdots \\ &\quad + \alpha_3(\mathrm{A}_1\cos\omega_1 \mathrm{t} + \mathrm{A}_2 cos\omega_2 \mathrm{t})^3 + \cdots \\ &= \left(\alpha_1 A_1 + \frac{3}{4}\alpha_3 A_1^3 + \frac{3}{2}\alpha_3 A_1 A_2^2\right) cos\omega_1 t + \left(\alpha_1 A_2 + \frac{3}{4}\alpha_3 A_2^3 + \frac{3}{2}\alpha_3 A_1 A_1^2\right) cos\omega_2 t + \\ &\quad \frac{3\alpha_3 A_1^2 A_2}{4}\cos(2\omega_1 - \omega_2)\, t + \frac{3\alpha_3 A_1 A_2^2}{4}\cos(2\omega_2 - \omega_1)\, t + \cdots \end{aligned} \qquad \text{(Eq. 21)}$$

We notice that the last two terms exhibit frequencies that do not belong to any harmonic of ω_1 or ω_2. We call this phenomenon as Intermodulation (IM). Among the IM products, the 3rd-order IM (IM3) products $\frac{3\alpha_3 A_1^2 A_2}{4}\cos(2\omega_1 - \omega_2)\, t$ and $\frac{3\alpha_3 A_1 A_2^2}{4}\cos(2\omega_2 - \omega_1)\, t$ are very important. If ω_1 and ω_2 are adjacent channels, the IM3 at $2\omega_1 - \omega_2$ and $2\omega_2 - \omega_1$ are adjacent with ω_1 and ω_2 respectively (cf. Fig.17 (a)).

In the case that two strong interferers exist at ω_1 and ω_2 while the desired signal is transmitting around $2\omega_1 - \omega_2$ or $2\omega_2 - \omega_1$, the two IM3 products will drop exactly into the band of interest at the output of the nonlinear block (cf. Fig.1.17 (b)). They increase rapidly as the input interferer amplitudes A_1 and A_2 increase and finally they will have the same strength as the fundamentals.

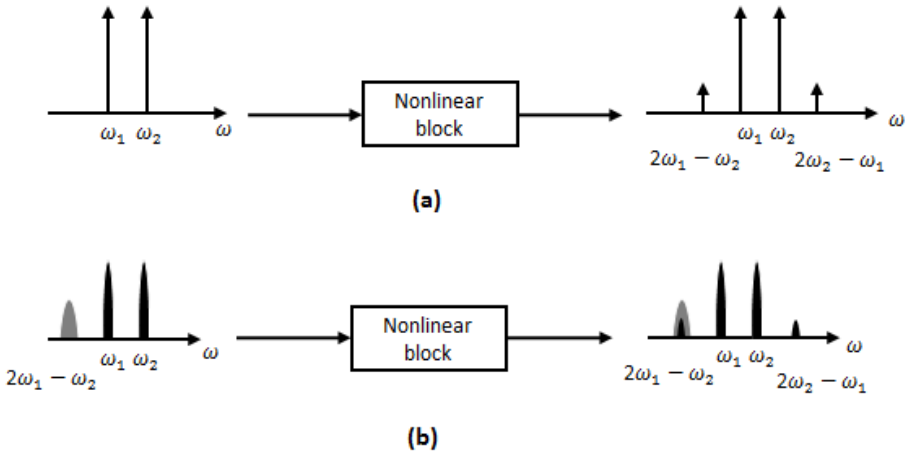


Fig. 1.17: (a) Two-tone intermodulation and (b) example of signal corruption due to IM3

In RF design, we use the 3rd intercept point (IP3) to measure the IM3 products. As shown in Figures 1.18 and 1.19, the IP3 is measured using dual tone input signal. We apply a two tones test with two signals which have the same amplitude A, and draw the linear fundamental amplitude $\alpha_1 A$ and IM3 amplitudes in a log-log scale with the development of the input amplitude A.

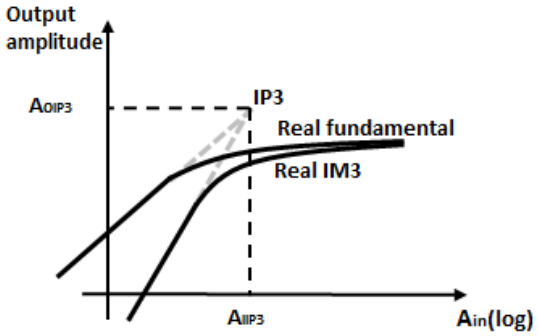


Fig. 1.18: IP3 definition

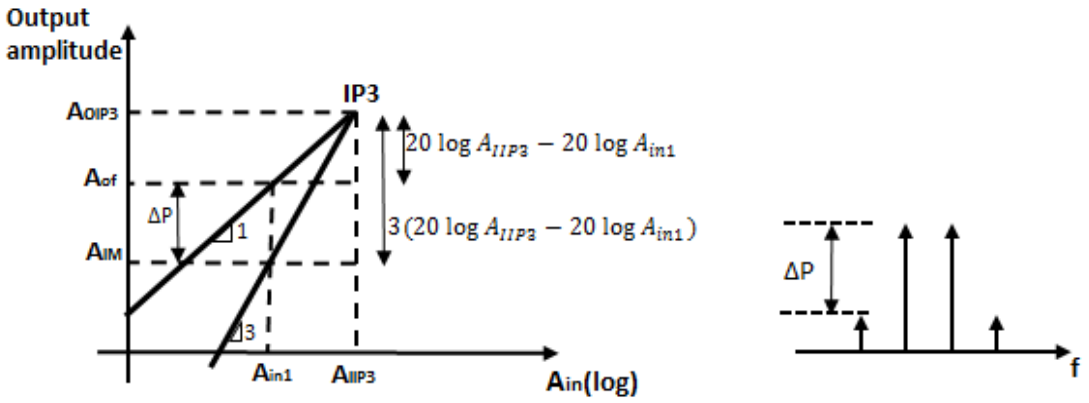


Fig. 1.19: IP3 measurement with two-tone test

The two curves will cross at a virtual point when:

$$|\alpha_1 A_{IIP3}| = \left|\frac{3}{4}\alpha_3 A_{IIP3}^2\right|,\ A_{IIP3} = \sqrt{\frac{4}{3}\left|\frac{\alpha_1}{\alpha_3}\right|} \qquad \text{(Eq.22)}$$

where A_{IIP3} is the input IP3 and the vertical-axis projection A_{OIP3} is the output IP3. As the actual output curves of the fundamental and IM3 are compressed (continuous lines in Figure 1.18), the IP3 doesn't actually exist. The measurement should be carried out with very small A which is located in the linear zone. Since the IP3 is normally far away, it is not easy to choose the input amplitude values and the intervals of the iterations. A little offset of the line slope leads to enormous deviation of IP3 values.

Another easier method is based on a hypothesis that the IM3 product equals to the fundamental output when the input reaches A_{IIP3}. If we applicate a smaller input A_{in}, it will generate a fundamental with an amplitude A_{of} which is $20\log A_{IIP3} - 20\log A_{in1}$ lower than Aoip3 since the fundamental curve slope in log is 1. This A_{in} will generate IM amplitude A_{IM} which is $3(20\log A_{IIP3} - 20\log A_{in1})$ lower than A_{OIP3}. We let the difference between the fundamental output amplitude Aof and the IM product amplitude A_{IM} is ΔP, given by Eq. 23.

$$\Delta P = 20\log A_{of} - 20\log A_{IM} = 2(20\log A_{IIP3} - 20\log A_{in1}),$$

$$20\log A_{IIP3} = \frac{\Delta P}{2} + 20\log A_{in1} \qquad \text{(Eq.23)}$$

This method is much more intuitive and easier to applicate but still with cautions that input amplitude should be far away from A_{IIP3}. This is easy to verify during the IP3 measurement. We only have to overlap the nonlinear output (Eq.21) with a hypothetic linear fundamental output coming from the input multiplied with α_1. The measurement result will be correct only if the fundamental tones superposed which indicates that the input amplitude is small enough to ignore the high order terms in Eq.21, and the output is in the linear zone, as shown in Figure 1.20.

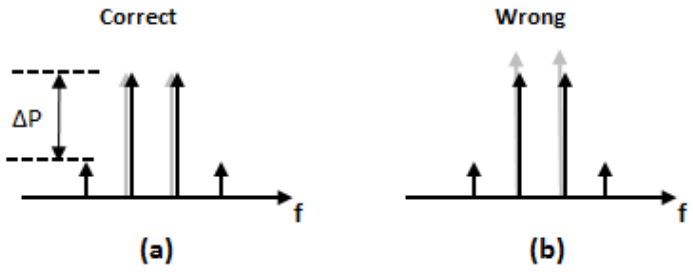


Fig. 1.20: ΔP measurement with situation: (a) correct and (b) wrong

Cascaded IIP3

At the end of the nonlinearity measurement introduction, we give a brief conclusion about the cascaded IIP3 measurement.

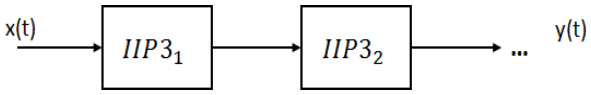


Fig. 1.21: Cascaded nonlinear blocks

Consider a chain of nonlinear blocks as shown in Figure 1.21. Each block has a measured $A_{IIP3,n}$ which has a relationship with the chain A_{ip3} as

$$\frac{1}{A_{IP3}^2} \approx \frac{1}{A_{IP3,1}^2} + \frac{\alpha_1^2}{A_{IP3,2}^2} + \frac{\alpha_1^2\beta_1^2}{A_{IP3,3}^2} + \cdots . \qquad \text{(Eq.24)}$$

This result also comes from a two tones analysis.

Non-linear modelling using equivalent BB model

We have introduced BB equivalent model in section 2.2. However two problems needed to be solved: How to propagate nonlinear effect between RF components, which cause harmonics in addition to the principal modulated signal? And how to manage adjacent frequencies, a problem met when simulating frequency offset or IIP3 test. To solve the first

problem, Vasilevski [12] proposed to represent a modulated signal more accurately taking into the DC and 2 harmonics into account, as described in Equation 25, extending the BB equivalent representation to (DC, I_1, I_2, I_3, Q_1, Q_2, Q_3).

$$x(t) = DC + I_1\, cos(w_c t) + I_2\, cos(2w_c t) + I_3\, cos(3w_c t) + \cdots$$

$$Q_1\, sin(w_c t) + Q_2\, sin(2w_c t) + Q_3\, sin(3w_c t) \qquad \text{(Eq.25)}$$

The second problem is about adjacent frequencies representation [12]. According to Eq. (26):

$$cos(w_c t + \Delta w t) = cos(\Delta w t)\, cos(w_c t) + sin(\Delta w t)\, sin(w_c t) \qquad \text{(Eq.26)}$$

the offset Δw applied to the signal frequency w_c will be distributed to amplitudes I1 and Q1. In other words, when a signal $x(t) = A\, cos(w_c t)$ is represented by (DC=0, I_1=A, I_2=0, I_3=0, Q_1=0, Q_2=0, Q_3=0), a signal $x(t) = Acos[(w_c + \Delta w)t]$ will be represented by (DC=0, I_1=Acos(Δwt), I_2=0, I_3=0, Q_1= Asin(Δwt), Q_2=0, Q_3=0).

That is why a high frequency signal with low frequency variations has to be oversampled to enlarge the band of each harmonic represented by BB equivalent samples. The simulation speed-up is closely related to the bandwidth around the harmonics that is simulated, as represented in Figure 1.22. The maximal timestep is the value for which, the bandwidths meet each over in the middle ($\Delta w = w_c/2$). In this case, all frequencies can be simulated from 0 to $3.5w_c$, but the simulation speed will be similar to a conventional non-BB equivalent simulation. The adapted timestep is determined by the type of simulation considering or not some adjacent frequencies, or frequencies deviations.

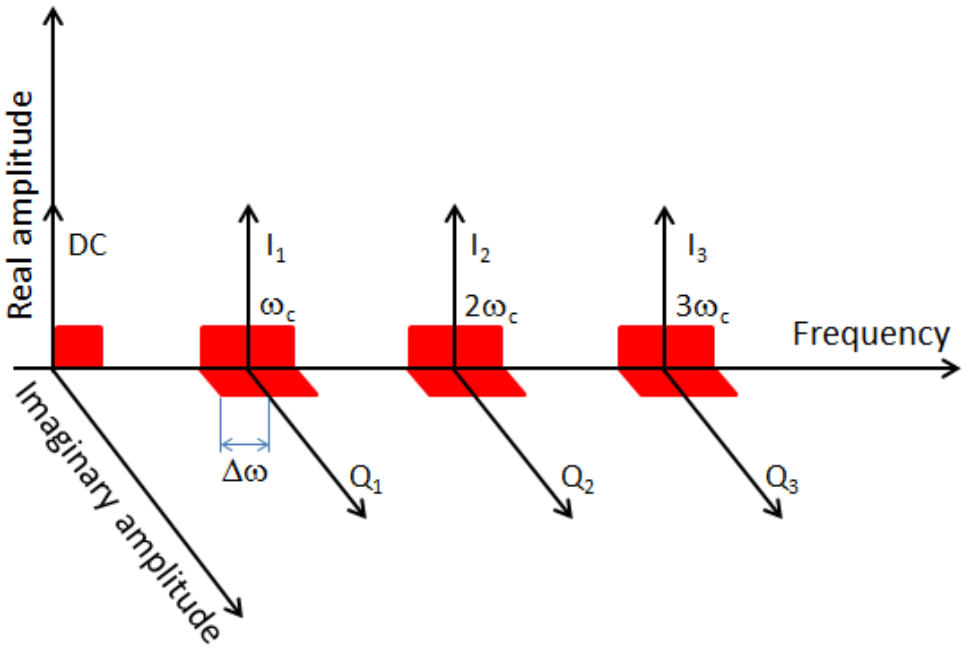


Fig. 1.22: Non-linear signal in BB equivalent modelling

2.4.4. Average power

In a previous work [13], we have developed a hierarchical library associated with various simulators that can be used in a single platform, called TrustMe-ViP, which enables a unique simulation framework and full model interoperability. Such platform is dedicated to complex SoC design, such as trusted personal devices where cost and time-to-market are very important constraints. Multi-language description (SystemC, Simulink, VHDL-AMS and SPICE netlist), multi-engine (Modelsim, Matlab, Eldo and EldoRF), multi-domain (MAC, BB, RF) and hierarchical description levels (system, behavioral, structural and transistor) are used to simulate a BT transmitter [13]. A dedicated RF design and verification platform was coupled with a SystemC emulation environment, which allowed the simulation of a full Bluetooth transmission/reception flow. This starts from the protocol emulation and the sent data generation, and then the modulation and the RF front-end are modeled and optimized. Added to that, the PA power consumption was estimated, based on relatively accurate simulation results, and subsequently correlated to high-level system parameters (such as the BER). The results present a brief overview of the countless possibilities of this power-aware modeling methodology, put together with a transparent plug-and-play simulation framework.

The same approach will be used in this work, where the power consumption of all block of a BLE transceiver is extracted from Spice simulations or measurements, which is important to realize accurate energy estimation in our global system simulation. Then, the average power consumption is considered as a generic parameter of a high-level SystemC-AMS model.

2.4.5. Conclusion

We reminded in this section the main characteristics of a RF front-end. The parameters of gain, input and output impedance, bandwidth, noise, non-linearities and power consumption, which we presented, will be extracted from simulation Spice or from measures. They will be used as generic parameters of the high level models (written in SystemC-AMS language), which we shall develop in the following chapter, the building blocks of a BLE transceiver. Before describing these models, we remind to the following paragraph the BLE standard, then we shall present the languages SystemC and SystemC-AMS.

3. Bluetooth Low Energy standard

3.1. Introduction

This section will firstly give a brief and general description about BlueTooth (BT) and BlueTooth Low Energy (BLE) technology. Then we will compare their system architectures. Since we have chosen BLE RF transceiver as the modeling objective, the BLE link layer, which is located just above the PHY, will be detailed (modes, BLE frame form introduction). Finally we will focus on the BLE PHY detail includes the wireless communication frequency bands, RF transceiver specification and modulation characteristics.

3.2. BlueTooth and BLE

BlueTooth (BT) is a wireless personal area network technology which is managed by Bluetooth Special Interest Group (SIG) and was defined as an IEEE standard since its first ratified version in 2002 [14]. BT is aimed at replacing cables between devices to the short-range wireless radio channel communication. With the low power consumption and low cost advantages, BT is widely used and developed over the years. It is nowadays one of the most attached wireless communication technology for the mobile communication equipment as it is completed through versions to support larger and varies files transmission with alternative data rates.

Bluetooth Low Energy (BLE) was completed as part of the BT Core specification version 4.0 in 2010 by Bluetooth SIG. It is aimed at very low power communication with shorter data frames, fast connection, maximized idle time and low peak transmit or receive power. It exchanges data in short bursts which fits well the needs of the low throughput devices. With these characteristics, BLE is used by lots of devices that are powered by small, coin-cell batteries such as watches and toys. Other devices such as sports and fitness, health care, keyboards and mice, beacons, wearables and entertainment devices are enhanced by this version of the technology. Within the framework of the CoCoE project, we suggest using this standard to develop sensors' network to monitor the electric consumption in buildings.

3.3. BLE system architecture

BLE architecture includes two types of chips which are BLE single-mode devices and Bluetooth v4.0 dual-mode devices. The BLE single-mode device contains only the BLE stack which can communicate with other BLE devices and also the BLE dual-mode devices when they are using the BLE technology part in their architecture. The BLE dual-mode devices are merged with the Classic BT stack and the BLE stack, this makes them capable to communicate with both of the Classic BT and BLE devices.

The architecture of a single-mode BLE stack which is shown below in Figure 1.23 can be divided as controller, host and application (cf. Table 1.1).

Table 1.1: BLE system construction

Application	Host	Controller
The application is the highest layer and the one responsible for containing the logic, user interface, and data handling of everything related to the actual use-case that the application implements. The architecture of an application is highly dependent on each particular implementation.	• Generic Access Profile (GAP); • Generic Attribute Profile (GATT); • Logical Link Control and Adaptation Protocol (L2CAP); • Attribute Protocol (ATT); • Security Manager (SM); • Host Controller Interface (HCI), Host side.	• Host Controller Interface (HCI), Controller side; • Link Layer (LL); • Physical Layer (PHY);

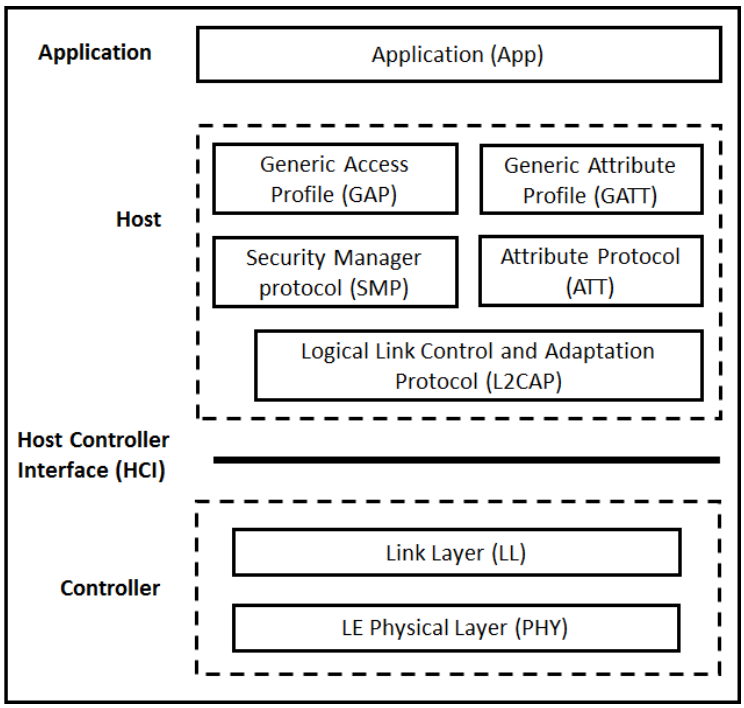


Fig. 1.23: BLE single mode protocol stack

This work is focused on the PHY transceiver modeling, and it is very important to understand the functionalities and operation of LL part. In the latter chapter, we will realize a global simulation of a BLE complete system which is integrated with the refined PHY transceiver models. Thus the modeling of the interface between LL and the transceiver is absolutely necessary. The understanding of LL functionalities is very important for the interface modeling and also for the realization of the global simulation scenario. The following introduction will be focused only on the LL layer and the PHY layer.

3.4. BlueTooth and BLE link layer

BLE link layer (LL) is placed between the PHY and the L2CAP (Logical Link Control and Adaptation Protocol) layer. It is in charge of controlling, negotiating and establishing the links, selecting frequencies to transmit data, supporting different topologies and supporting various ways for exchanging data [15].

The mainly functionality of LL contains:

- Preamble, Access Address, and air protocol framing
- CRC (Cyclic Redundancy Check) generation and verification
- Data whitening
- Random number generation
- AES (Advanced Encryption Standard) encryption

3.4.1. LL states

LL operates in the way of the state machine shown in Figure 1.24 with the five states as Standby State, Advertising State, Scanning State, Initiating State and Connection State. The description of each state is listed in Table 1.2.

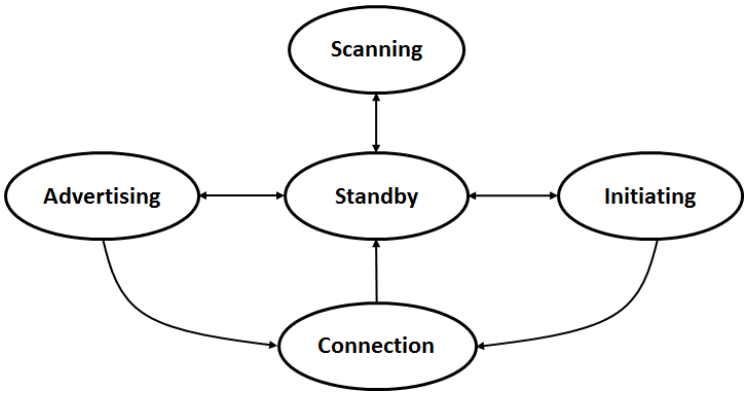


Fig. 1.24: Link Layer State Machine

Table 1.2: Description of the five LL states

Standby State	This is the default idle state. There is no packet exchanging in this state. A BLE device can come back to this state from any other LL states.
Advertising State	This is the state where LL exchanges the advertising packets. A BLE device can enter this state from the standby state when it decides to start advertising and it becomes an Advertiser in this state.
Scanning State	This is the state where LL listens on the advertising channel and waits the packets come from the Advertiser. It can also communicate with Advertiser for having additional information. A BLE device can enter this state from the standby state when it decides to start scanning and it becomes a Scanner in this state.
Initiating State	This is the state where LL listens on the advertising channel and waits the packets from Advertiser and then it responds to those packets when it initiates a BLE connection. A BLE device can enter this state from the standby state when the Scanner decides to initiate a connection with the Advertiser. LL in this stated called an Initiator.
Connection State	This is the state where the BLE devices are connected, and it can be entered from the initiating state (which turns the LL to a Master) or from the advertising state (which turns the LL to a Slave).

3.4.2. BLE Packet format

The link layer has unique packet format which is shown in Figure 1.25 includes Preamble, Access Address (AA), Protocol Data Unit (PDU) and CRC.

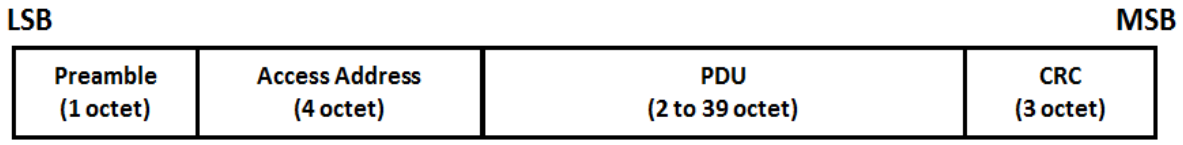


Fig. 1.25: Link layer packet format

The 8-bit Preamble is used in the receiver to perform frequency synchronization, symbol timing estimation, and Automatic Gain Control (AGC) training. It is 10101010b for Advertising channel packets, and for the data channel packet, it is 10101010b if the AA LSB is 0 and 01010101b if the AA LSB is 1.

AA for all advertising channel packets shall be 10001110100010011011111011010110b (0x8E89BED6). In data channel packets, it is generated with restrictions in BT core4 [14] by initiator and used in a connection request.

Protocol Date Unit (PDU) format depends on the nature (advertising or data) of the packet. The 24-bit **Cyclic Redundancy Check** (CRC) is calculated over the PDU after the PDU encryption if the PDU is encrypted by using a 24-bit CRC polynomial as $x^{24} + x^{10} + x^9 + x^6 + x^4 + x^3 + x + 1$.

During the bits transmission, the long zeros or long ones make bits transmission error-prone. BLE use **Data whitening** technique to avoid this kind of long sequences by using a 7-bit Data whitening polynomial as $x^7 + x^4 + 1$. This process is applied on the PDU and CRC parts so it is carried out after the CRC generation.

LL **device filtering policy** as a way to save the power is worth to mention. LL uses the device filtering mechanism to focus on the devices it is interested in by using a "white list" which is configured by the Host. It contains a set of device addresses inside of this list, and it is used to filter out the non-desired Advertisers, Scanners and Initiators. LL won't respond to any request packet from the devices whose device address is not included in this list, and these packets will just be ignored.

3.4.3. LL state machine operations

When the LL is in a non-Standby state, it will exchange a corresponded type of PDUs in different events. Now we will introduce the procedures and different events in procedures of advertising, scanning, initiating and connection to give a general idea of the LL operation. All the details about the timing will be ignored since it is not our study focus point.

Advertising & Scanning

When LL is in the adverting state, it has to use advertising channels to send and receive advertising PDUs in 4 types of advertising events as in the Table 1.3 which is dragged out from BT 4.0 specification.

When LL is in the scanning state, it can perform either a passive scanning where it can only listen to certain advertising channel and receive packets but not send packets, or an active scanning where it can receive packets and respond by sending SCAN_REQ PDU to an advertiser to ask for additional information when it receives ADV_IND or ADV_SCAN_IND PDUs from that advertiser.

Table 1.3: Advertising events and corresponding PDUs [16]

Advertising Event Type	PDU used in this advertising event type
1. Connectable Undirected Event	ADV_IND
2. Connectable Directed Event	ADV_DIRECT_IND
3. Non-connectable Undirected Event	ADV_NONCONN_IND
4. Scannable Undirected State	ADV_SCAN_IND

When an advertiser uses the Type 1 or the Type 4 events in Table 1.3, it permits a scanner to require additional information by sending SCAN_REQ PDU. When the advertiser uses these two types of event, it sends ADV_IND PDU or ADV_SCAN_IND PDU respectively to the advertising channels and then it listen on the same channel for the respond from a scanner while a scanner can respond to these packets with a SCAN_REQ PDU to obtain additional information. Figure 1.26 shows an example of the procedure of Type 1 event used advertising with a SCAN_REQ PDU reception from an active scanning. In this figure, an advertiser is sending ADV_IND to each of the three advertising channels, and when it is in the 38 channel, it receives a SCAN_REQ which can pass the advertising policy, then it responds this request by sending a SCAN_RSP in the same channel.

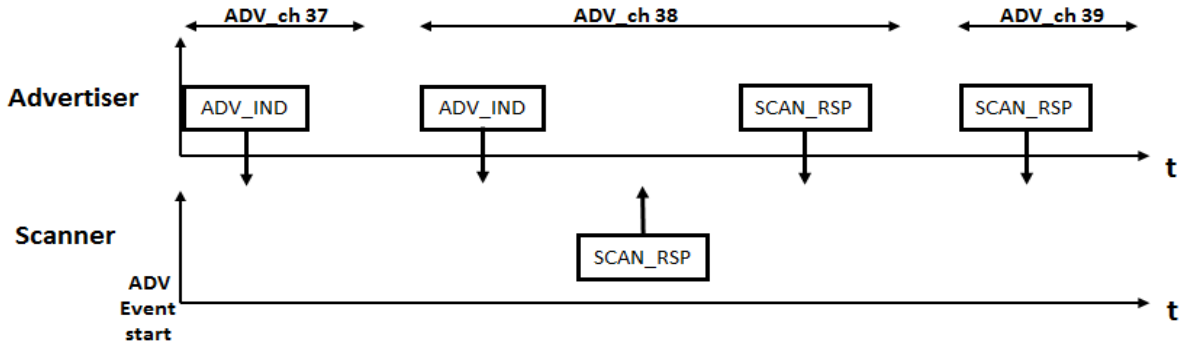


Fig. 1.26: Advertising event example with a scanner sending SCAN_REQ in channel 38

When an advertiser uses the Type 3 event in Table 1.3, it sends the non-connectable advertising indication (ADV_NONCONN_IND) to the advertising channels but without listening. Thus a scanner can obtain information by receiving this packet but it can't respond to it.

Advertising & Initiating

When LL is in the initiating state, it opens a ScanWindow to listen on each advertising channel. When it receive ADV_IND PDU or ADV_DIRECT_IND PDU which is allowed by the initiator filter policy, it will respond with a CONNECT_REQ PDU to require a connection. The initiator enters to connection state after sending the CONNECT_REQ PDU.

When an advertiser uses the Type 1 or Type 2 events in Table 1.3 and it receives a CONNECT_REQ PDU, it will become a Slave by entering the connection state with come conditions. Figure 1.27 shows an example of the procedure of Type 1 event used advertising with a CONNECT_REQ PDU reception from an initiator.

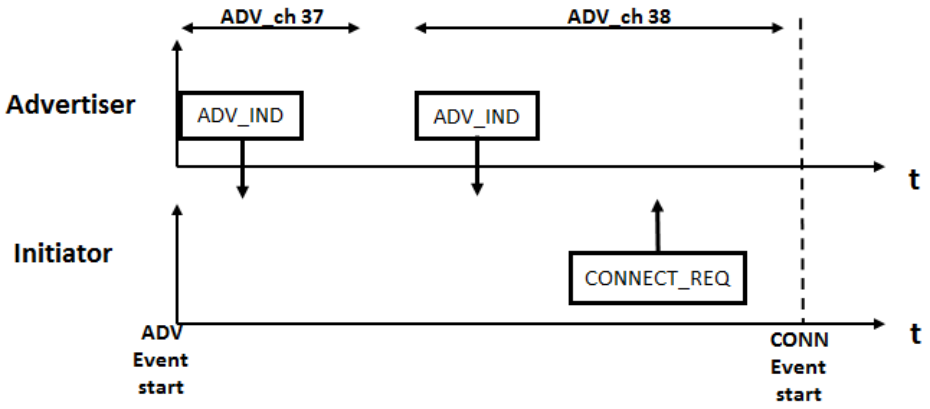


Fig. 1.27: Advertising event example with an initiator sending CONNECT_REQ in channel 38

Connections

As we have talked before, an LL goes to the connection state when it sends a CONNECT_REQ PDU as an initiator or when it receives a CONNECT_REQ PDU as an advertiser. When an initiator enters the connection state it becomes a Master, and when an advertiser enters this state it becomes a Slave. They are allowed to communicate with Data channel PDUs. When an LL enters this state the connection is created. Then to establish this connection, there should be at least one successful Data packet reception.

The Master and the Slave choose a channel index and use it to exchange Data packets during the same connection event. The timing in connection state can be generally determined by two parameters:

• Connection event interval (connInterval): It defines the interval between two consecutive connection events;

• Connection slave latency (connSlaveLatency): It defines the number of connection events that the Slave has the right to skip but still with connection.

Both of the master and the slave use supervision timer detect a possible connection loss at any time during a connection. This timer counts the maximum time interval between two valid packet receptions and if this interval exceeds the predefined maximum interval, a supervision timeout will be created which will make an LL exit the connection state and go back to the standby state.

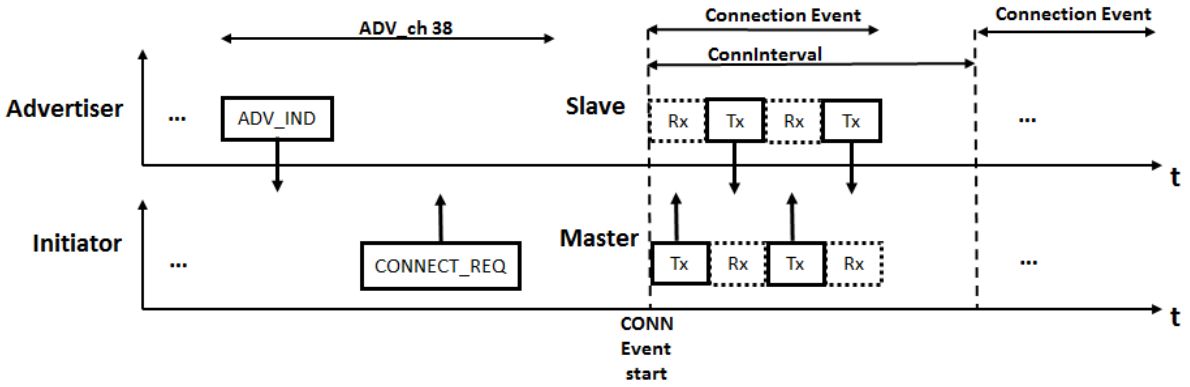


Fig. 1.28: Two LLs enter to connection event from advertising and initiating respectively

During the packets exchange in a connection event, both Master and Slave indicate whether they have more data to send by using the MD bit in the packet header (cf. Fig. 1.28). If both of them don't have any more data to send, the Master will close the event and the Slave won't listen on the channel after its packets sending. If either of them or both still have data to send, the Master should maintain the event and the Slave has to listen on the channel after its packet sending.

3.5. BLE Physical layer description

BLE physical (PHY) layer is the part where the transceiver is presented with which the analog signals are modulated or demodulated and exchanged through physical channels.

BLE radio operates in the 2.4 GHz Industrial, Scientific and Medical band (ISM) which is an unlicensed frequency band. The complete frequency range used by BLE is from 2.400 to 2.4835 GHz. In this range, BLE uses 40 channels which have the central frequencies as 2402 + k * 2 MHz, where k is from 0 to 39 (cf. Fig. 1.29). The three channels with indexes of 37, 38 and 39 are used as advertising channels to exchange broadcast data and to establish BLE connections. The rest 37 channels are data channels which are used to exchange data during BLE connections.

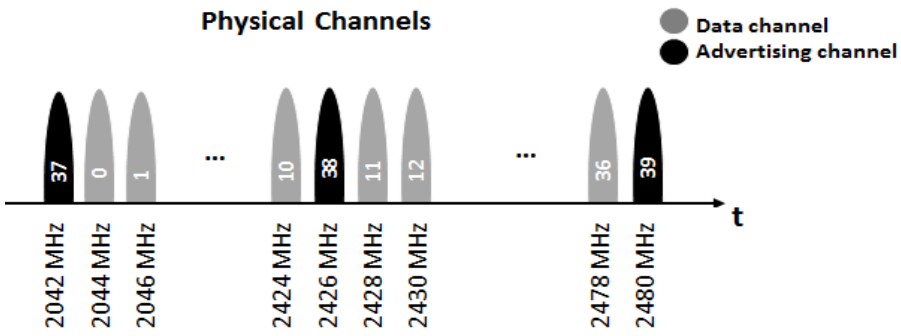


Fig. 1.29: Physical channel index and corresponding center frequency

The communication in the ISM band could have lots of interference because there are other wireless technologies as WLAN and NFC, etc., which operate also in this band. So BLE uses frequency hopping (FH) technique to combat interference and fading. FH means that the RF signals are switched rapidly to different channel according to a pseudorandom sequence of the channel index.

The BLE modulation is the Gaussian Frequency Shift Keying (GFSK) with a modulation rate of 1 Mbit/s. The bandwidth-bit period product is fixed as 0.5 and the modulation index should be between 0.45 to 0.55. A binary one is represented by a positive frequency deviation and a zero is represented by a negative frequency deviation. GFSK principle will be introduced in the next subsection.

BLE is defined for a transmitter output power between 0.01mW to 10mW. The receiver sensitivity level is set to be less than or equal to -70 dBm, for a 0.1% BER.

3.6. GFSK modulation

In the passband transmission, Frequency Shift Keying (FSK) is a digital modulation which converts the digital baseband signals (0 or 1) to passband frequency modulated signals. In the case of the binary FSK for example, two frequencies f_1 and f_2 are used to represent binary information 0s and 1s respectively. Then the FSK modulated signal can be described by Eq.27, and wave form is like shown in Figure 1.30.

$$e_{2FSK} = \begin{cases} Acos(\omega_1 t + \varphi_n) & symbol\ "1" \\ Acos(\omega_2 t + \theta_n) & symbol\ "0" \end{cases} \quad \text{(Eq.27)}$$

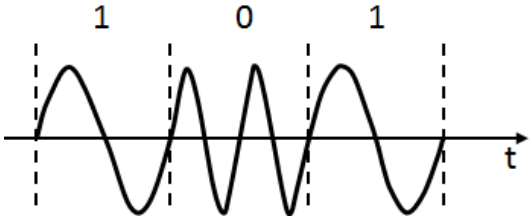


Fig. 1.30: FSK example for transmitting "101"

FSK is widely used is because that the FSK modulated signals have better noise immunity in comparing with the amplitude modulated signals, and FSK modulated systems are less complicated for the demodulation.

Gaussian (GFSK) modulation is the Gaussian filter applied FSK modulation. GFSK uses a Gaussian filter to smooth the beginning of each digital symbol before the frequency modulation procedure. The Gaussian filtering avoids the high frequencies due to the switching, and thus reduces the signal spectral bandwidth which will reduce the adjacent channel interference.

3.7. Direct and indirect angle modulation

It is worth to mention that there are two extension types of angle modulation, which are the indirect FM and the indirect PM.

A PM RF signal can be described as

$$s_{fm}(t) = Acos(\omega_c t + K_p s_B(t)), \quad \text{(Eq.28)}$$

where $K_p$is proportionally constant and $s_B(t)$ is the baseband signal.

While an FM RF signal is given by Eq. 29.

$$s_{FM}(t) = Acos(\omega_c t + \int_0^t K_f s_B(\tau) d\tau), \quad \text{(Eq.29)}$$

where K_f is also a proportionally constant.

As the frequency and the phase can be converted into each other with integration and derivation, the angle modulation can be extended to four methods as shown in Figure 1.31. They are the direct and the indirect FM, and as the direct and the indirect PM.

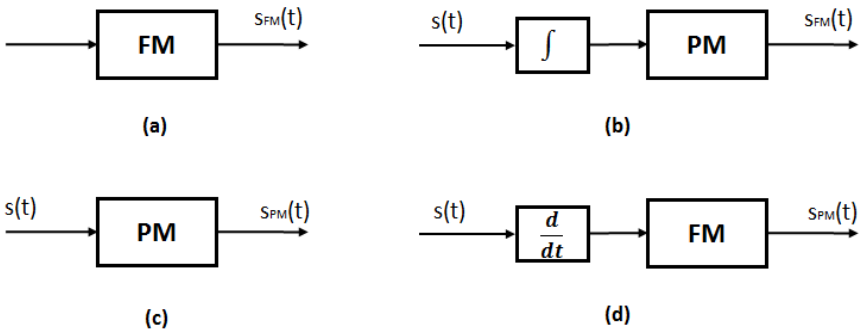


Fig. 1.31: Angle modulations: (a) Direct FM; (b) Indirect FM; (c) Direct PM; (d) Indirect PM

If we firstly differentiate the baseband signal and then realize an FM, we will obtain a PM signal (cf. Fig. 1.31 (d)). This method is called indirect PM. And if we firstly integrate the baseband then realize a PM, we will obtain an FM signal (cf. Fig. 1.31 (b)). This method is called indirect FM.

4. SystemC toolset : Mixed signal simulation tools

4.1. Introduction

In communication system design, there are various modeling tools and languages. Designs will be efficient in time and reliability only if we choose the ones that correspond to our design requirements. In this subsection, we will give brief introductions of some common

modeling languages and tools, then we will focus on SystemC tools introductions which is chosen for this work, and the interface establishments between the models of different domains described in SystemC tools [16].

4.2. SystemC

SystemC is a set of C++ classes and macros for system level modeling, design and verification [17]. On one hand, SystemC has semantic similarities to hardware description languages such as VHDL and Verilog. On the other hand, it allows modeling software components by using directly C/C++ code by inheriting all of the advantages of C++ like the rich language constructs and data types which always make hardware designers reduce the time and efforts of work. SystemC supports the design in different abstraction levels so that designers can model either a large system in a very high abstraction level or an analog timed block with the predefined electrical primitives. Basic SystemC is based on a Discrete Event (DE) model of computation in which a scheduler maintains a time-sorted list of events, executes them accordingly and updates signals before increasing the simulation time. SystemC-DE can be used for event-driven models. In particular, two specific models of computation are available, i.e., RTL to model clocked behavior as well as bit-accurate signals and TLM to model interactions through function calls.

4.3. SystemC-AMS

SystemC provides an extension to model analog and mixed signal (AMS) components; in particular, three specific models of computation are available [17]:

- Timed Data Flow (TDF). It allows to model a system as a sequence of blocks producing and consuming data samples with a user-defined rate. TDF can be used for time-driven models. It permits to describe also discretized continuous-time systems, e.g., by using Laplace transfer functions.
- Linear Signal Flow (LSF). It supports natively continuous-time models by combining primitives such as addition, gain, integration, derivative, and delay operators. Discretization should not be addressed explicitly by the user.
- Electrical Linear Network (ELN). It supports natively continuous-time models by combining predefined linear electrical components, such as resistors and capacitors. Their characteristics are used to describe the continuous-time relationships between voltage and current levels.

4.4. Verilog-AMS, VHDL-AMS

Verilog-AMS (respectively VHDL-AMS) is the analog and mixed signal extension of the Verilog HDL (respectively VHDL). It is used to model blocks with behavioral or

structural modules encapsulate with ports and additional parameters. It simulates the analog behaviors with conservative solution which is the Kirchhoff's Potential and Flow Laws.

Previous works were led to the EpOC laboratory on the hierarchical modelling of building blocks of a RF transceiver by using the language VHDL-AMS [18], as well as on the simulation of a BT communication [19].

4.5. Matlab

Matlab is a powerful worldwide used tool developed by MathWorks. It is based on C language and is usually used for matrix calculations, functions plotting, ect., and it can interface with programs written in C, C++, Java and Python. Applying for embedded system design, Matlab is also a data flow simulator. It is able to model RF analog systems containing non-idealities and the additional Simulink packet is easy and intuitive for models description. But it is limited in simulation speed for RF system simulation, not to mention a complete wireless communication system.

The coupling between Matlab and VHDL-AMS [20] already allowed to realize simulations system multi-engine and multi-level of BT communications within the EpOC laboratory: thesis of Benjamin Nicolle [19], Alexandre Lewicki [21] and Lucas Alves Da Silva [13]. An article of synthesis, summarizing all these works, was published in Microelectronics Journal [22].

4.6. Mixing SystemC MoCs

Previous sections introduced the modeling strategies for communications. Even when only SystemC is considered, different models of computation can be used. In particular, this section aims at presenting how to mix TDF, LSF, RTL, and TLM MoCs.

Figure 1.32 (a) gives an overview of the architecture used to interface a TDF component with an RTL component. In this case, since SystemC Discrete Event processes are not allowed within TDF modules, an adjunctive TDF module has been inserted to replicate the TDF interface of the component as an RTL interface by using *sca_tdf::sca_de* ports. This architecture allows to maintain unchanged the TDF implementation of the component.

Figure 1.32 (b) gives an overview of the architecture used to interface an LSF component with an RTL component.

Even if LSF descriptions are used to specify continuous time components (i.e., ODE), they allow the presence of any classic SystemC process (i.e., SC_THREADs or SC_METHODs) obeying to the standard Discrete Event model of computation used by basic SystemC. Moreover, input and output ports of LSF clusters can be written or read by discrete processes.

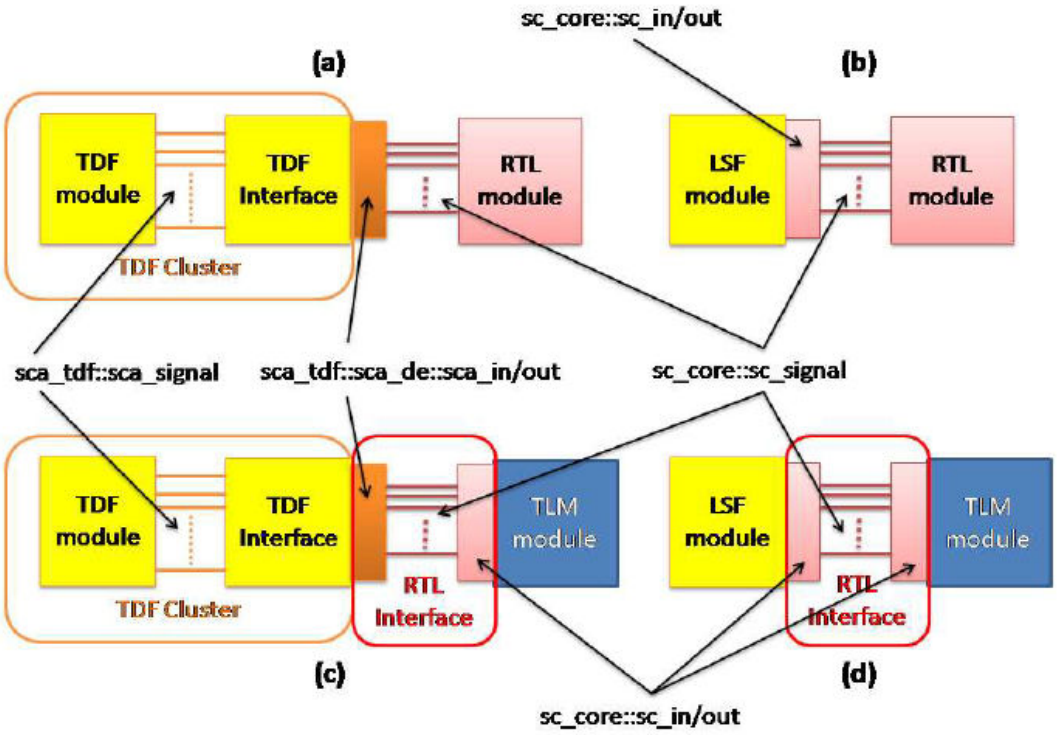


Fig. 1.32: SystemC MoCs connections: (a) TDF-RTL (b) LSF-RTL (c) TDF-TLM (d) LSF-TLM

Therefore, the LSF module can be directly enriched with an RTL interface reproducing exactly the same input and output ports provided by the LSF cluster. SystemC discrete processes are inserted into the LSF module for taking care of writing the values received by the RTL input ports into the LSF input ports and from the LSF outputs into the RTL outputs. The processes providing inputs to the LSF clusters are sensitive to the RTL inputs ports.

SystemC-AMS TDF and LSF components cannot be directly connected with TLM components. Therefore, an intermediate RTL interface is needed. The interconnection between LSF and RTL or between TDF and RTL has been already described. The RTL interface acts as transactor [23] with respect to the TLM block. The *b_transport* function, used to receive transactions from the TLM part of the system, will "unpack" data and write them into the output ports of the RTL interface. The function responsible for sending data to the TLM part of the system will fill TLM payload with data coming from the RTL interface. Figures 1.32 (c) and 1.32 (d) give an overview of the architectures used to interface a TDF or LSF component with a TLM component respectively.

4.7. Conclusion

In this work, we choose to use the SystemC toolset to model a complete BLE system, because it corresponds to the requirements to simulate such an entire system. These tools support high level models which can be fast simulated. Moreover these tools cover every domain in a wireless system, and make it easy to interface different domains. They are

efficient for a complete simulation platform with the Meet-in-the-Middle method which will be talked later. As shown in Figure 1.33, the combination of SCAMS and the BB equivalent modeling method is enough to describe RF blocks; the analog system modeling can be handled with SystemC-AMS and all the digital system is modeled with SystemC/SystemC-TLM.

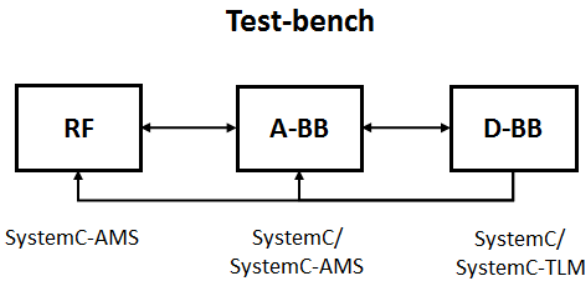


Fig. 1.33: Simulation test-bench with chosen tools of different domains

5. Conclusion

In this chapter, we have presented some challenges for modeling and simulating a complete wireless system which contains different domains. According to these problems, we have chosen the modeling method and techniques for this work after talking and comparing the existed modeling methods and tools. In addition, we have talked about the principles of the specifications of RF blocks modeling which is useful for the block refinement.

As shown in Figure 33, the analog and RF part of the transceiver will be modelled using SystemC-AMS and the Digital part with SystemC/TLM. This approach allows us to re-use the testbenches (use cases of BLE applications) developed by RivieraWaves using SystemC-TLM.

In the next chapter, we will talk about a BLE transceiver modeling, refinement and verification by using SystemC-AMS. We focus on the architecture developed by RivieraWaves.

Chapter II – BLE transceiver modelling

1. Introduction

This section will give an introduction of a defined BLE transceiver architecture. The objective of this chapter is to model this transceiver in system-level with refinements of the key RF parameters talked in the section 2.4 (chapter 1).

A complete BLE transceiver can be functionally separated into 4 blocks as the modulation, the demodulation, the transmitter and the receiver. As shown in Figure 2.1, this transceiver contains GFSK modulator and demodulator for the BLE technique. For the RF architecture, we have chosen a Zero-IF topology for the RF transmitter and a Low-IF (intermediate frequency) topology for the RF receiver [24]. It contains also the most important interfaces between the transceiver and the LL (Link Layer) which are the transmitted (Tx) and received (Rx) data, the power selection signal and channel selection signal [25]. However, these interface signals will be talked about in the next chapter.

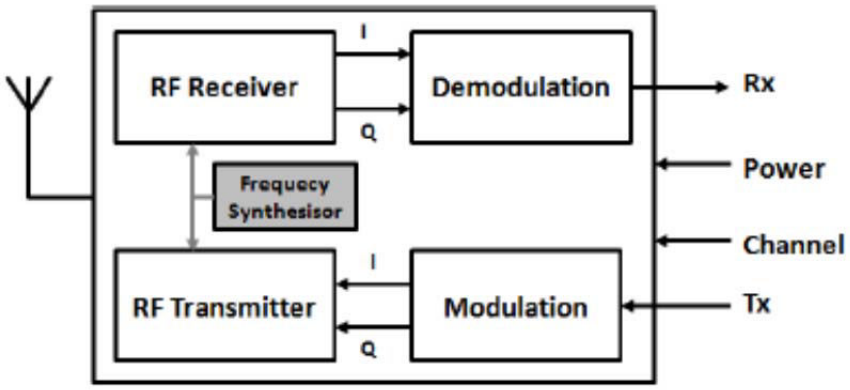


Fig. 2.1: BLE top level block diagram

BLE transceiver uses GFSK modulation. As we have talked in the last chapter, GFSK is a FSK with a Gaussian pulse shaping before the frequency modulation. So as shown in the transmitter (cf. Fig. 2.2), the output of the Gaussian filter (GF) is the mapped frequency deviations. It was also presented in the last chapter that for the FSK, we can use direct and indirect modulation techniques.

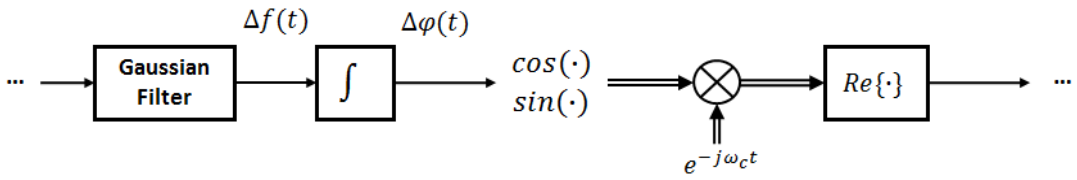


Fig. 2.2: Simplified Tx with indirect GFSK modulation

This BLE transmitter uses a narrow band FM architecture based on the indirect FSK modulation as shown in Figure 2.2. The frequency deviations are converted to instantaneous phases and transmitted as a PM signal.

In this work, we focus on the Rx performance analysis and optimization, so the Tx blocks modeling will be ideal. The RF receiver is a Low-IF type architecture with an intermediate frequency of -1 MHz. The received RF signal is firstly down converted to IF. Since we choose a negative IF, the signal in the Rx process is complex. The IF signal is filtered by a complex band-pass filter and digitized by a band-pass $\Delta\Sigma$ ADC (Analog to Digital Converter). The down conversion to the baseband is processed by a Digital Signal Processor (DSP), as depicted in Figure 2.3.

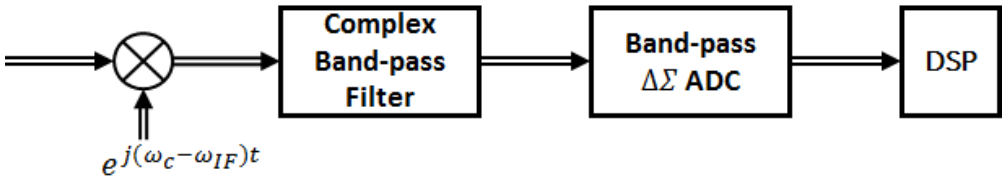


Fig. 2.3: Simplified Rx Front-end

First, the DSP is running the demodulation, which is the inverse process of the modulation. After the signal is brought back to baseband, $\Delta\varphi(t)$ is obtained. It is then converted to $\Delta f(t)$ with the $tan(\cdot)$ operation. The details of the transceiver modeling will be presented in the next section. It begins with the functional introduction and modeling of the whole transceiver in Fig. 2.1. The following part is the very important Rx front-end modeling. It contains the functional modeling, following by the RF refinement and some verifications. Since the RF blocks are modeling in the system-level abstraction, for several blocks it exists more than one way to realize the modeling. In this case, we will talk and compare the different solutions.

2. Transceiver modelling and simulation

2.1. GFSK modulation modelling

2.1.1. GFSK modulator architecture

The GFSK modulation process is shown in Fig. 2.4, which begins with a bits generator who sends “0”s or “1”s with a data rate of 1 MHz. These bits are signed by a Binary Frequency Shift Keying (BFSK) block which uses two symbols with a phase difference of 180° to represent binary bits as “-1” for “0” and “+1” for “1”. The following step is the indirect frequency modulation process. The signed baseband data is shaped by a Gaussian filter (GF) which generates 13 samples output during each input data period, which makes the

output data rate up to 13 MHz. The GF output is considered as instantaneous frequency deviation as $\Delta f(t)$ and it is integrated as instantaneous phase deviation as $\Delta\varphi(t)$ by a digital integrator. This phase signal is then mapped as $e^{j\Delta\varphi(t)}$ by using quadrature transmission as $\cos[\Delta\varphi(t)]$ for inphase path (path I) and $\sin[\Delta\varphi(t)]$ for the quadrature phase path (path Q). The last step is to convert the digital signal to analog with DAC. Since the transmitter is modeled as ideal, the DAC is simplified to a convertor which reserves its input with a fixed number of bits resolution output, without considering any noise.

The 3 blocks as the "Bits Generator", the "Gaussian Filter" and the "Digital integrator" are very important to understand the modulation process. They will be introduced in details in the rest of the subsection.

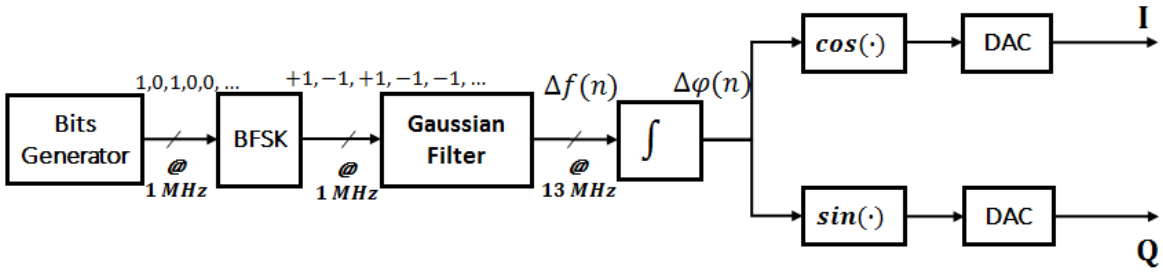


Fig. 2.4: GFSK modulation block diagram

2.1.2. Bits generator

The bits generator provides BLE packets as shown in Figure 2.5 which ignores the CRC (Cyclic Redundancy Check) part because the CRC check is not necessary for the receiver performance estimation.

Each packet transaction begins with a calling of the function *DE* to firstly store the BLE packet into a stack as "p" with the type "uint8_t" (unsigned byte: 8 bits), e.g. the preamble part of the packet is one byte (where 0x55 equals "01010101" with the most left bit as LSB). It is thus stored as the first cell of the stack "p", and it is followed by the first byte of AA which is 0x29, etc. Each byte is mapped in order to an 8 bits buffer "x", then send one by one bit with a time interval of 1 μs which corresponds to 1 MHz data rate.

SC-AMS system needs a time interval initiation in the first block. The situation here is like that we connect SC and SC-AMS domains where an RTL interface is needed, which is as defined here as:

tx.write(x[i]);
wait(1.0, sc_core::SC_US).

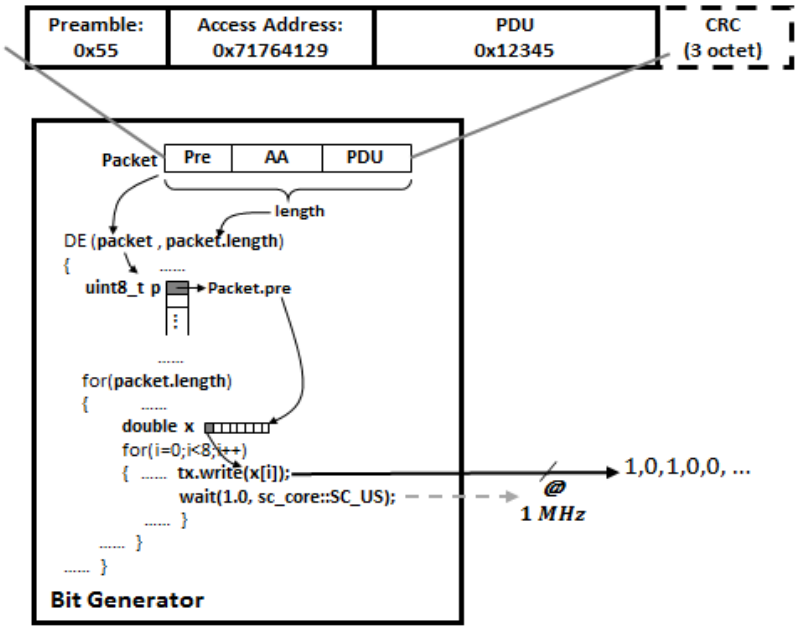


Fig. 2.5: The determined BLE packet used to generate bits

2.1.3. Gaussian filter

The Gaussian filter (GF) is realized with an oversampling of 13 [25]. So, the output data rate of this block becomes 13 MHz from 1 MHz due to the Gaussian pulse shaping process as shown in Figure 2.7 (a).

GF is a 39 tabs FIR filter with the 39 coefficients generated by using the Marcum's function [26]. They are stored as a look-up table (LUT) $c[k]$ where k is from 0 to 38. The GF shape is shown in Fig. 2.6.

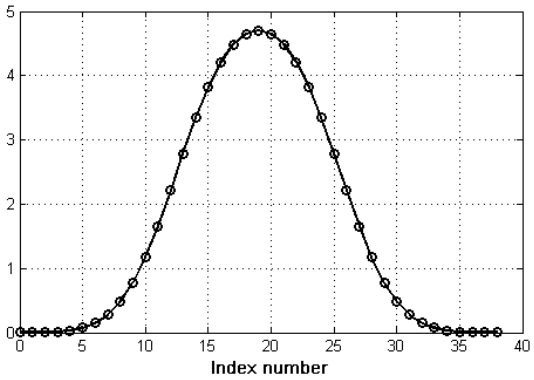


Fig. 2.6: Gaussian filter form with 39 coefficients

A complete GF shaped pulse is used for 3 µs long mapping which means that each 1 µs in the output is overlapped by different parts from 3 Gaussian pulses: the ramp-down 13 samples of the previous pulse $g[n-1]$, the middle 13 samples of the instantaneous pulse $g[n]$ and the ramp-up 13 samples of the successive pulse $g[n+1]$.

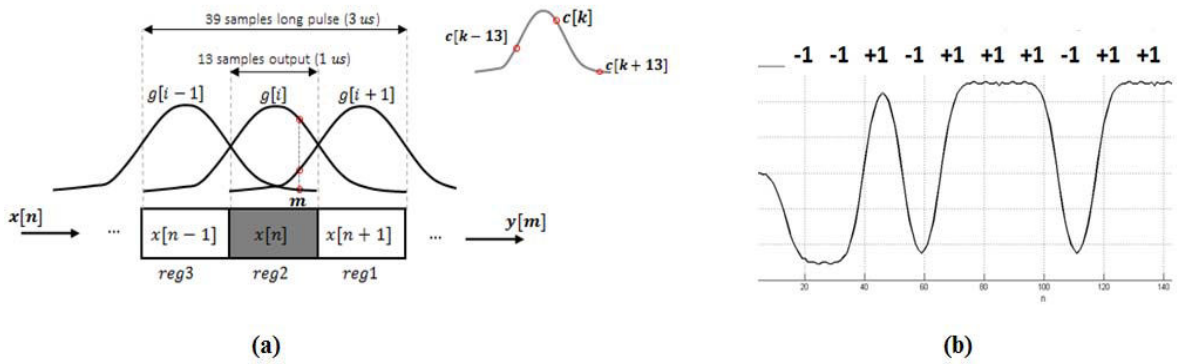


Fig. 2.7: (a) GF pulse shaping process; (b) Example of GF output wave

Assuming that the period of 1 μs with dark grey corresponds to the instantaneous input x[n], it generates 13 values at the output during this period. Observing one moment of m in this period which corresponds to the output instant, there are three overlapped samples as g[i][m], g[i-1][m] and g[i+1][m], which come from three different pulses. Since the Gaussian pulse is symmetric, it turns out that the three values coming from different pulses can be represented with three values of one single pulse with an interval of 13 samples from each other.. The value g[i][m] corresponds to c[k]*x[n], where k=m mod 39. Then g[i-1][m] equals to c[k+13]*x[n-1] and g[i+1][m] equals to c[k-13]*x[n+1]. So it needs only three registers to memorize the signs of the three GF input symbols that one Gaussian pulse occupies.

An example of the GF output is shown in Figure 2.7 (b). The overlapping effect can be seen from this wave that the maximum and minimum values appear when there are continuous "-1"s or "+1"s.

2.1.4. Digital integrator

The GF output indicates the instantaneous frequency deviation $\Delta f(n)$. It will be integrated as the instantaneous phase shift $\Delta\varphi(n)$ by using an IIR digital integrator which has a transfer function as:

$$H(z) = \frac{1}{1-z^{-1}} \qquad \text{(Eq.1)}$$

This transfer function can be implemented as shown in Fig. 2.8, which leads to a system with the input $x[n]$ and output $y[n]$ having a relationship given by:

$$y[n] = x[n] + y[n-1] \qquad \text{(Eq.2)}$$

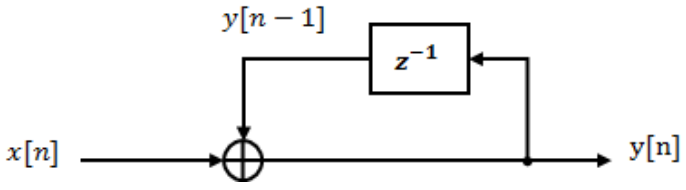


Fig. 2.8: Implementation of the digital integrator

Temporally it is easy to realize this block by using a variable to represent $y[n-1]$, which is the accumulation of all the "old" $y[n]$.

2.2. Demodulator modelling

2.2.1. Demodulator architecture

The demodulator architecture, depicted in Figure 2.9, is realized by using a series of operations reciprocal to those of the modulation process.

An "arctangent" operation is used to detect the deviated phase. This signal $\Delta\varphi(k)$ corresponds to the output of the "Frequency/Phase conversion" block in the modulation process. Then a derivation is used to get the deviated frequency information. In the end, we need also a correlator to finally find out the binary bits sent by the "Bits Generator" in the beginning of Tx. More details can be found in the following sub sections.

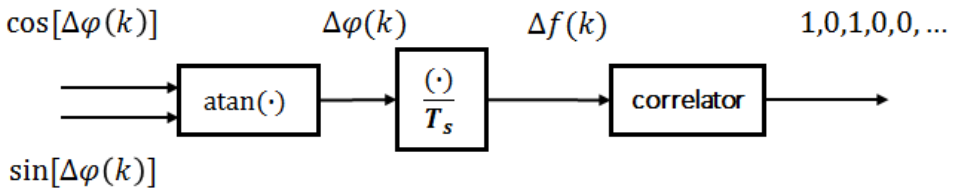


Fig. 2.9: Demodulation block diagram

2.2.2. Arctangent operation and derivation

The operation of arctangent used to get the modulated $\Delta\varphi(t)$ is modeled with a direct application of the C++ function *double atan2 (double x, double y)*. The result of this block (Eq. 3) corresponds to the digital integrator output in the transmitter.

$$\Delta\varphi(t) = arctan(\frac{sin\Delta\varphi(t)}{cos\Delta\varphi(t)}) \qquad \text{(Eq.3)}$$

To get the frequency deviation information, we need a derivation of the phase $\Delta\varphi(k)$. A digital derivation at a certain moment gives the difference between the instantaneous value and the previous one, which is $y(k) = x(k) - x(k-1)$.

If the frame contains a long period of "0"s or "1"s, the Arctan operation will produce a phase jump when the phase is greater than π or lower than –π. So, we have added a corrector block in order to take into account this phenomenon. This block replaces the sample value, which introduces a 2π jump, by the previous one.

2.2.3. Correlator

The system needs a correlator to detect BT Access Code (AC) or BLE Access Address (AA). It allows the system to determine when to sample the incoming oversampled stream in order to down-sample the Rx Data stream and recover the data at 1 MHz. The correlation results in a BT/BLE system is used to extract the PDU data in a received packet by detecting its AC/AA with the synchronization word. The correlator structure is shown in Figure 2.10.

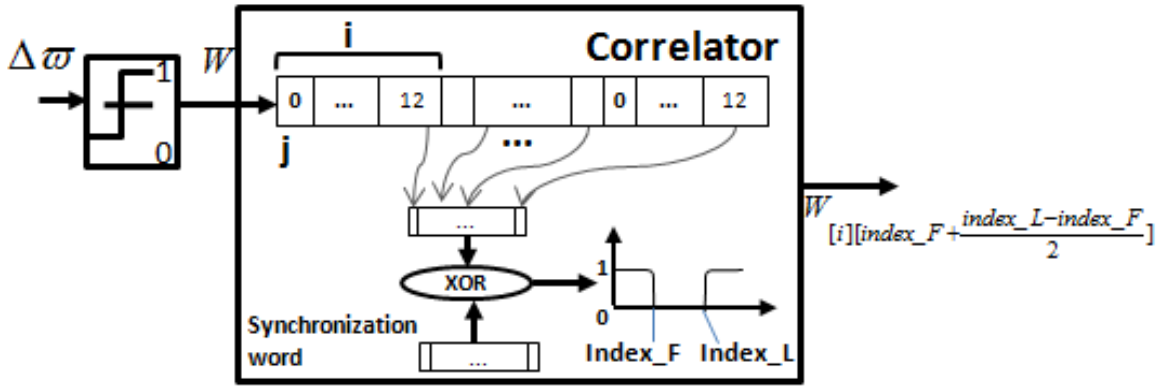


Fig. 2.10: Correlator process

The sign of the Differential output Δf is converted into logic "0" and logic "1" and stored into a buffer. The synchronization word is compared with the incoming W oversampled data stream using the i to $j + 13$ bits, where i and j represent respectively the index of the group of bits every 1 μs and the index of each bit during 1 µs. The comparison (XOR) calculates the number of error found on each clock cycles. The stream is considered as matching the synchronization word when no errors are detected.

During the comparison the number of errors is processed with the detection of the first instant of matching detection (indicated by index_F), and the last instant of matching detection (index_L). Those first and last index values are used to calculate the optimal sampling instant using the following formula:

$$\text{Opt\_Samp} = \text{index\_F} + \frac{(\text{index\_L} - \text{index\_F})}{2} \qquad \text{(Eq.4)}$$

Once the optimal sampling instant is detected, the complete packet can be extracted with all the indexes matching the Eq.5 (Preamble and AA) and Eq.6 (PDU data).

$$\text{Packet}[i] = \text{RxStream}[\text{Opt\_Samp} - i * 13] \quad (i \text{ is from 0 to 39}) \qquad \text{(Eq.5)}$$

$$\text{Packet}[40 + i] = \text{RxStream}[\text{Opt\_Samp} + i * 13]$$
$$(i \text{ is from 40 to the PDU length} - 1) \qquad \text{(Eq.6)}$$

2.3. Analog/RF Tx modelling

2.3.1. Tx architecture

The RF transmitter is composed of a digital-to-analog conversion (DAC), Low-pass filtering and an RF conversion (cf. Fig. 2.11). The modeling of the DAC can be realized by repeating the digital samples (hold function). This process causes spectrum aliasing in the frequency domain, which requires the Low-Pass Filters (LPF) to remove the harmonics before the RF signal conversion with the local oscillators.

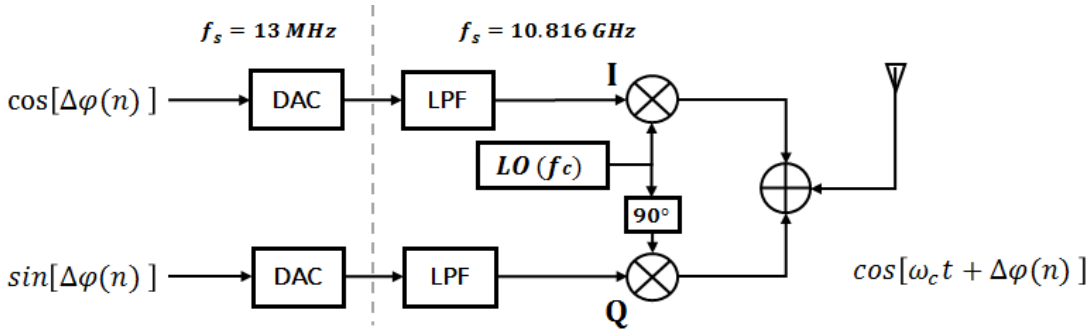


Fig. 2.11: RF transmitter block diagram

2.3.2. Digital-to-Analog convertor

The Digital-to-Analog convertor (DAC) is modelled as an interpolator with the output of a fixed number of bits accuracy (e.g. 6 bits). It works as repeating the same sample for N times where N is the number of oversampling. Theoretically, the system should work at a frequency at least 2 times greater than the highest data frequency (2.403 GHz in our case), to use enough simulation points to represent the signal information. Here we choose this frequency as $f_s = 64 \times 13 \times 13e6 = 10.816\ GHz$.

The interpolation doesn't exist in a real circuit. It is a choice to represent analog signals with less distortion. It is recommended to choose the adapted interpolation ratio for different frequency domain. Here in this step of our work is just an example to explain the modeling method. It will be optimized later in the RF refinement step where the system accuracy and the simulation time will be compared with this version.

2.3.3. Low pass filter modelling

The low pass filter (LPF) which is used to remove the harmonics before mixing with the local oscillators is a 3rd order Butterworth type filter which is used to reject the out of band signal. Its transfer function is given by Equation 7.

$$H(s) = \frac{\omega_p^3}{s^3 + 2\omega_p s^2 + 2\omega_p^2 s + \omega_p^3} \qquad \text{(Eq.7)}$$

where ω_p is the band width of the filter. SCAMS proposes Laplace Transfer Function (LTF) to describe a linear analog system transfer function by using the TDF MoC. In this way, a frequency response described block (filters) can be simulated in a timed system.

LTF can be used in two forms which are: numerator-denominator form and zero-pole form. As example, Eq.7 can be realized with the numerator-denominator form given by the code presented in the next page. In this code, "in" and "out" are the input and output of the filter respectively and the filter is represented by a defined LTF called "ltf1" which takes numerator and denominator values as parameters. The frequency response of this filter with a band width of 1 MHz is shown in Figure 2.12.

```
…… LTF SCAMS code …..
sca_vector<double> den, num;    // denominator & numerator
sca_tdf::sca_ltf_nd ltf1;
void processing()
{  out.write(ltf1(num, den, in.read()));  }
void initialize()
{  num(0) = wc * wc * wc;
   den(0) = wc * wc * wc;
   den(1) = 2 * wc * wc;
  den(2) = 2 * wc;
  den(3) = 1;  }
……
```

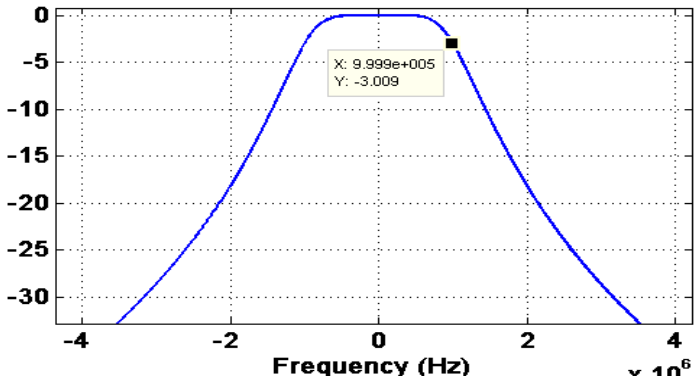


Fig. 2.12: Frequency response of the 3rd order Butterworth low pass filter

2.3.4. RF conversion

The local frequency in the transmitter is fixed at 2.402 GHz which means the system communicates in the channel 0. The oscillator generates two quadrature carriers as $\cos(\omega_c t)$ and $-\sin(\omega_c t)$, which are used to generate the final RF signal sent by the antenna by multiplying with $\cos(\Delta\varphi)$ and $\sin(\Delta\varphi)$:

$$\cos(\Delta\varphi)\cos(\omega_c t) - \sin(\Delta\varphi)\sin(\omega_c t) = \cos(\omega_c t + \Delta\varphi). \quad \text{(Eq.8)}$$

Figure 2.13 shows the spectrum of this signal. Its central frequency is around 2.4021GHz.

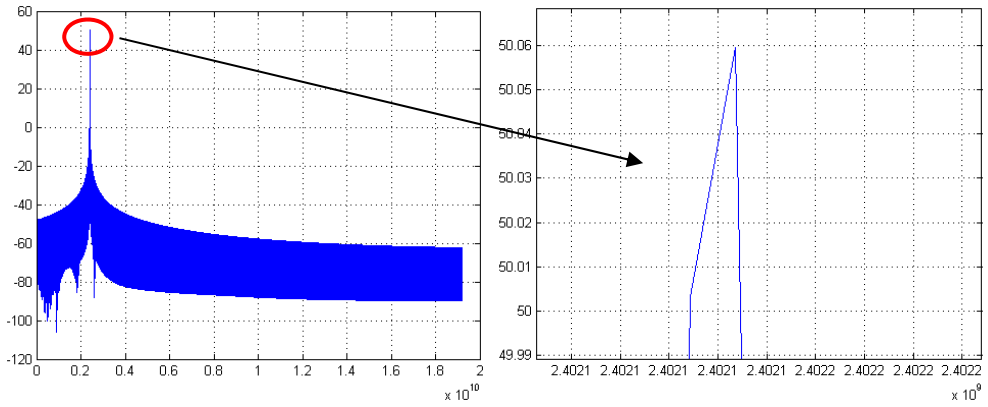


Fig. 2.13: Transmitter output spectrum

2.4. Analog/RF Rx functional modelling

2.4.1. Rx architecture

Following the introduction of RF transmitter modelling, in this section, the RF receiver architecture (cf. Fig. 2.14) will be presented. It is a defined low-IF architecture receiver chain with the intermediate frequency as $-1\ MHz$. It contains blocks of the low-noise amplifier (LNA), the mixer, the complex IF band pass filter (CXF1) and the Sigma-Delta (SD) type ADC. The working frequency of ADC is 52 MHz, so the simulation frequency is reduced to 52 MHz at the ADC input by an additional decimation of 208 which corresponds to the interpolation in the transmission. Since the highest simulation frequency is set to more than 10 GHz, the simulation will be very slow, so the first version of the RF receiver will be functional ideal modelling. As the ideal representation of LNA is a linear gain and ideal LO is a pair of carriers, we will focus in this subsection on the presentation of the blocks of the Low-IF down conversion, CXF1 and ADC ideal models.

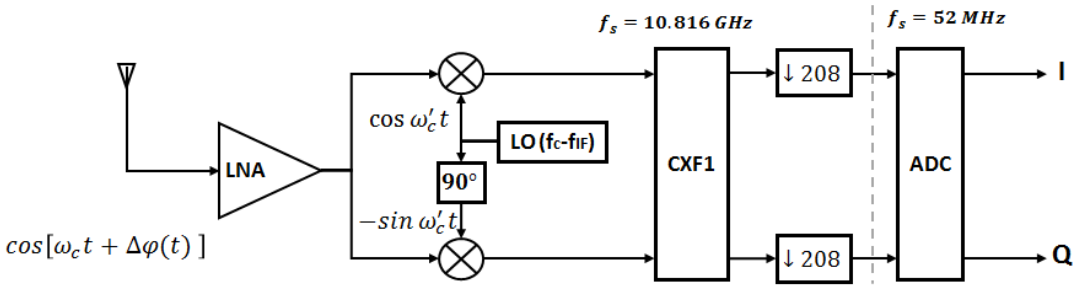


Fig. 2.14: RF receiver architecture

2.4.2. Low-IF down-conversion at -1MHz

To obtain the signal spectrum centered around $-1MHz$, we need to move the spectrum of the signal $\cos[w_c * t + \Delta\varphi(t)]$ towards left by multiplying $e^{-j*wc'*t}$, where $\omega_c' = 2\pi * (fc + 1\ MHz) = 2\pi \times 2.403\ GHz$.

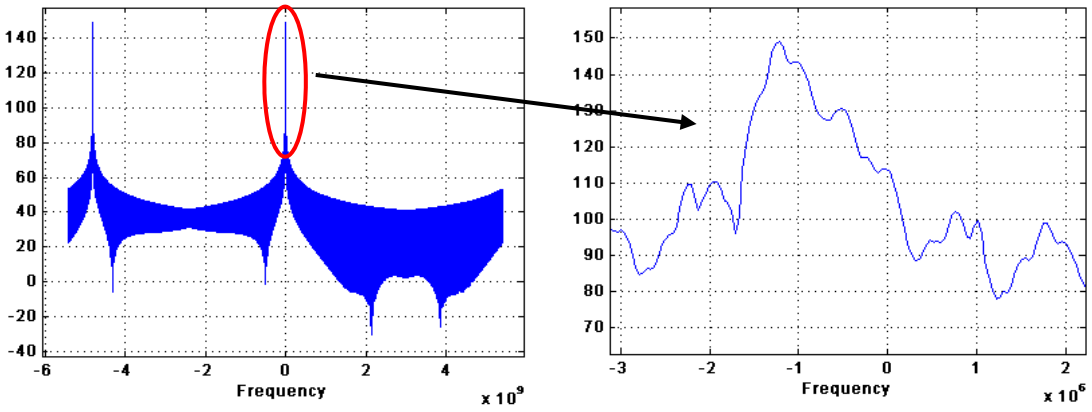


Fig. 2.15: Down-conversion at -1 MHz spectrum

Figure 2.15 shows the -2.403 GHz shifted spectrum. The right side spectrum is now centered at $-1\ MHz$ and the left one is centered at $-4.805\ GHz$ which is not desired and will be suppressed by the following block (CXF1).

2.4.3. Complex transfer function modelling

The complex filter (CXF1), following the mixer, has a frequency response given by Eq. 9.

$$H(\omega)_{cb} = \frac{G\omega_0^2}{[j(\omega-\omega_1)]^2+j\frac{(\omega-\omega_1)\omega_0}{Q}s+K\omega_0^2} \qquad \text{(Eq.9)}$$

where G is the filter gain, ω_o is the filter band width, ω_1 is the filter central frequency and Q is the quality factor. The subscript "cb" means "complex band-pass".

We introduce **two ways** to realize this filter transfer function modelling. The first way is to construct a direct architectural model.

We suppose that the system is stable and we let $s = j\omega$, then the filter transfer function given by Equation 9 becomes:

$$H(s) = \frac{G\omega_0^2}{s^2+\frac{\omega_0}{Q}s+K\omega_0^2} \qquad \text{(Eq.10)}$$

We change the frequency response in pole-zero form as

$$H(s) = \frac{A}{(s-a_1)(s-a_2)} \qquad \text{(Eq.11)}$$

This process can be modelled as two first order filter as shown in the figure below.

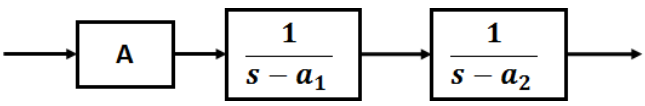


Fig. 2.16: Equivalent architecture of 2nd order filter

Then we can find the architecture of the filter as shown in Figure 2.17, where a, b, c and d are the real parts and imaginary parts of a_1 and a_2 respectively.

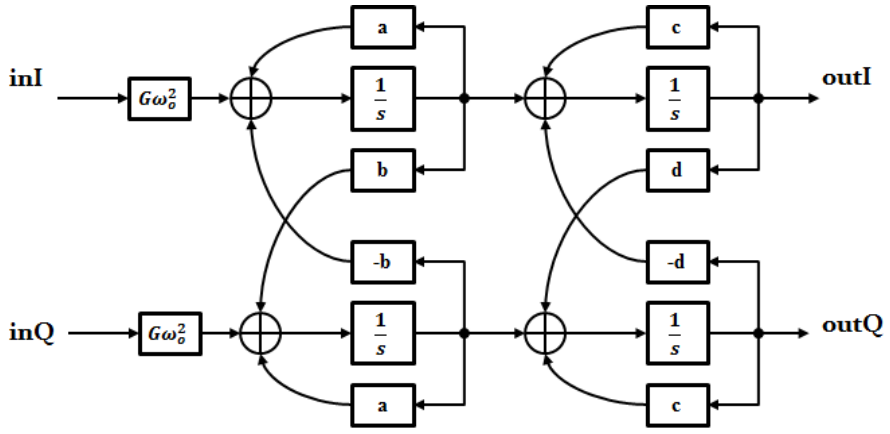


Fig. 2.17: Implementation of CXF1

This method is original and intuitive. But if we want to change the filter characteristics like the number of order and the number of coefficients, we have to modify the details inside of this block with manual work which makes the modelling process error-prone and time-consuming.

The second way to model the analog filter behavior is to use directly the frequency response equation by applying SCAMS proposed LTF as talked in the subsection of the demodulation.

By changing $\omega_1 = 0\,Hz$, the complex filter of Equation 9 becomes a low pass filter, which makes the frequency response as

$$\mathrm{H}(\omega)_{\mathrm{rl}} = \frac{\mathrm{G}\omega_0^2}{[\mathrm{j}\omega]^2+\mathrm{j}\omega\frac{\omega_0}{Q}\mathrm{s}+\mathrm{K}\omega_0^2} \qquad \text{(Eq.12)}$$

where subscript "rl" means "real low-pass". We suppose that the system is stable and we let $s = \mathrm{j}\omega$, then the filter transfer function can be obtained by Eq. 13.

$$\mathrm{H}(s)|_{s=j\omega} = \frac{\mathrm{G}\omega_0^2}{\mathrm{s}^2+\frac{\omega_0}{Q}\mathrm{s}+\mathrm{K}\omega_0^2}\,. \qquad \text{(Eq.13)}$$

Here we use a technique, where the principle is that instead of realizing a model of the complex filter directly, we model the LPF of equation 10 and we modulate the LPF input by a frequency $-\omega_1$ rad/s and the LPF output by ω_1 rad/s [27]. This procedure, presented in Figure 2.18, simplifies the modelling. Inside of the filter block, the complex input signal spectrum is firstly moved to right for $\frac{\omega_1}{2\pi}$ Hz with $e^{j\omega_1 t}$ which brings our desired signal to be centered at DC. Then this desired signal is selected by using the LPF obtained from the original $H(\omega)$ with $\omega_1 = 0\,Hz$. The final step is to shift the desired signal spectrum back to

ω_1 with $e^{-j\omega_1 t}$. This method is easy for a latter model parametrization which makes the analog blocks reusable. The CXF1 transfer function is drawing in Figure 2.19 and the output spectrum of this filter is shown in Figure 2.20. The spectrum around $4.804GHz$ is well attenuated by this filter.

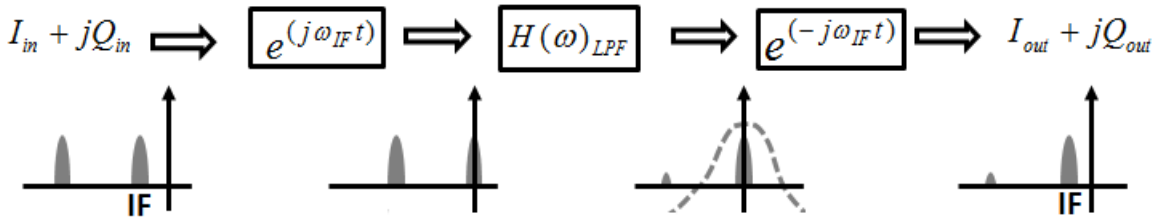


Fig. 2.18: CXF equivalent model

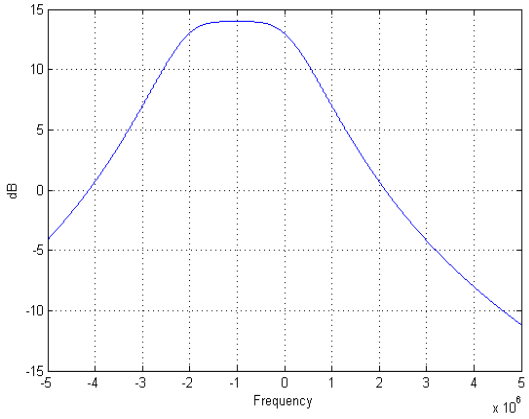


Fig. 2.19: Transfer function of CXF1

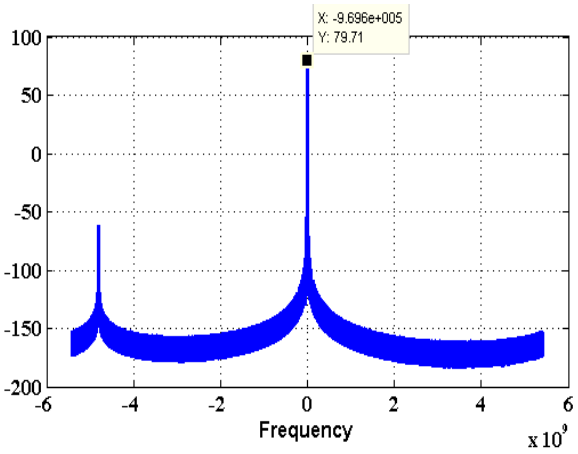


Fig. 2.20: CXF1 output spectrum

2.4.4. Sigma-Delta ADC

The ADC is a 3rd order quadrature band-pass digital sigma-delta type ADC with the output in 2 bits (-1, 0, -1) in I path and Q path. The working frequency is 52 MHz and the input signal is centered at -1 MHz and the desired band is 1 MHz.

The behavior of the complex band pass ΣΔ is easy to understand and designed with the same principle of CXF1. The design can be started from a low pass real ΣΔ transfer functions then shift the central frequency towards where we desire, which is the -1 MHz low-IF frequency in this system, by rotating this real ADC for an angular frequency of $\omega_\Delta = 2\pi \times \frac{-1\ MHz}{52\ MHz}$ radian. For the signal transfer function ($STF(z)$) and the noise transfer function ($NTF(z)$) of this ADC, it means that all the zeros and poles are multiplied by the term $e^{j\omega_\Delta}$.

For making the ΣΔ modulator working at its sampling frequency of 52 MHz, a down-sampling of $64 \times 13 \times 13\ MHz/52\ MHz\ = 208$ is done before the AD conversion.

As CXF1, the ADC is a complex block. It has a cross coupled structure for I and Q path. Figure 2.21 describes the implementation of this structure, where c_{ff} indicates the feedback facto. In this block, the coefficients and the signals are complex. The frequency response of this block is presented in Figure 2.22.

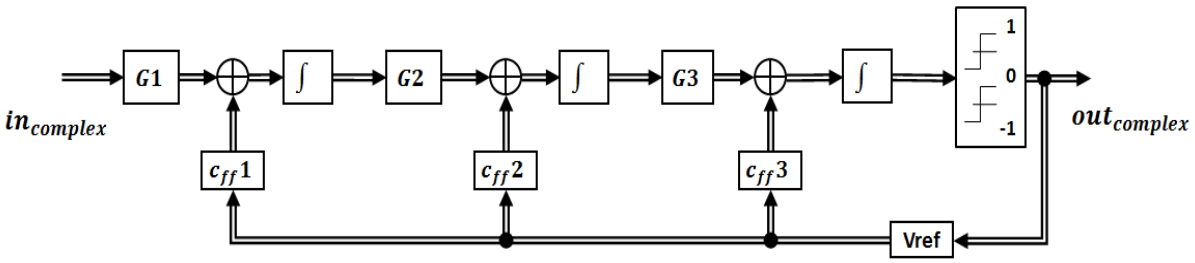


Fig. 2.21: ADC implementation

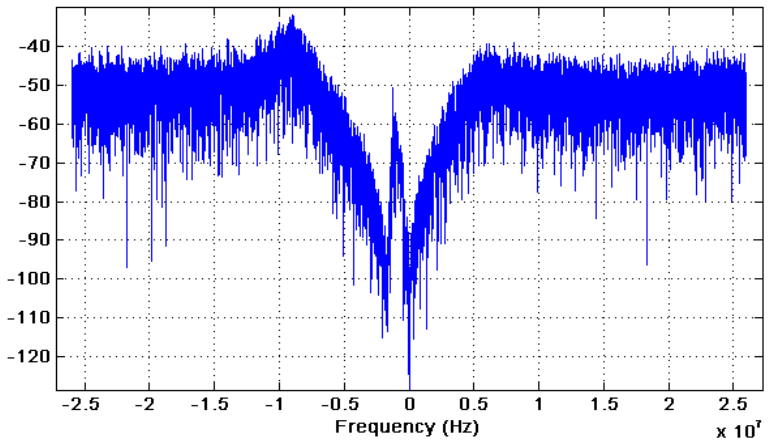


Fig. 2.22: ADC output spectrum

We can notice that the signal is centered at -1MHz, and the noise is shaping out of band (right and left sides due to the band pass behavior of the structure).

2.5. Receiver digital BB modelling

2.5.1. Rx BB architecture

In the digital baseband process as shown in Figure 2.23, the ADC output is firstly filtered by another bandpass complex filter (CXF2) to remove the out of band noise. The following blocks are two identical 4-tap filters which are used to attenuate the noise in the frequency bands. After the filtering, the signal data rate is reduced from 52 MHz to 13 MHz by a decimation of 4, and then the signal is brought from the low-IF to DC by a frequency shift block. Finally, the output band non-desired spectrum of the signal is removed by LPFs before entering to the demodulation process. In the following introduction of the modelling of each block, the LPF modelling, which is the last block in this part, will be presented firstly because it is a simple example with a typical architecture as a digital filtering.

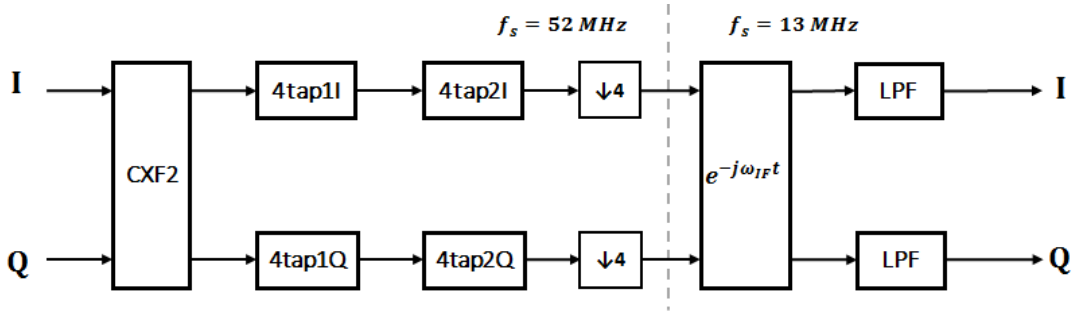


Fig.2. 23: Receiver digital baseband part architecture

2.5.2. Low pass filters

This is a FIR type digital filter which aims to reject the out of band noise due to the frequency shifting which is the last block before the demodulation, and it will be explained later in the RF receiver. This filter is obtained by using the "*fir1(N,Wn)*" function of Matlab where N is the number of the filter coefficients minus 1, and Wn is the filter cut-off frequency. In SystemC-AMS, there is no model for digital FIR filters realization, it is because we can easily realize FIR filter with the "for" loop of C/C++.

In the time domain, an N_{th} order FIR filter can be generally represented as a discrete-time system which has an input/output relationship given by Eq. 14.

$$y[n] = \sum_{i=0}^{i=N-1} c[i]x[n]z^{-i} \qquad \text{(Eq.14)}$$

where "c" contains N coefficients of the filter which are found with Matlab as talked above. This process can be implemented as shown in Figure 2.24.

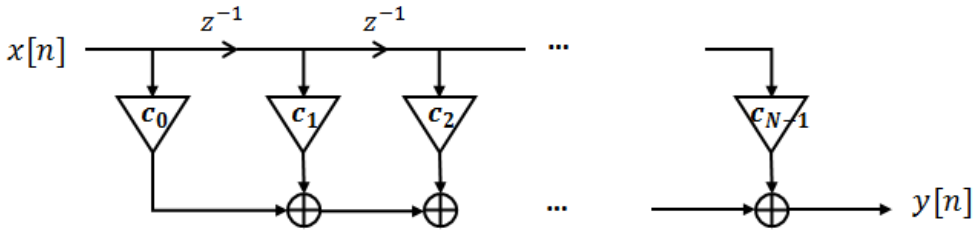


Fig. 2.24: Nth order FIR implementation

"z^{-1}" indicates one sample delay in time domain as talked earlier in the digital integrator. It is very simple to use the "for" loop of C/C++ to represent a series of delays. An example of the N_{th} order FIR filter in SC-AMS is shown below (cf. next page).

In this piece of code, "in" is the input of the filter which corresponds to "$x[n]$" (cf. Figure 2.24). The series of delays is realized by the vector "x" which is defined as a band of registers that store the "old" (or delayed) inputs in the "for" loop. The last value of "x" which is the instant input is taken in the end of the process because it is the only value not contained in any of the registers.

```
……
sca_util::sca_vector< double > x;
void processing()
{  ……
  double yout, sout;
  yout = 0.0;
  sout = 0.0;

  for( int i = N - 1; i > 0; i --)
  {  x(i) = x(i-1);
     yout += c(i)x(i);  }

  x(0) = in.read();
  sout = yout + c(0)x(0);
  …… }
……
```

The frequency response of an example with the cut-off frequency as 620 kHz is shown in Figure 2.25. The outputs of these filters corresponds to the "cos(·)" and "sin(·)" outputs in the modulation respectively.

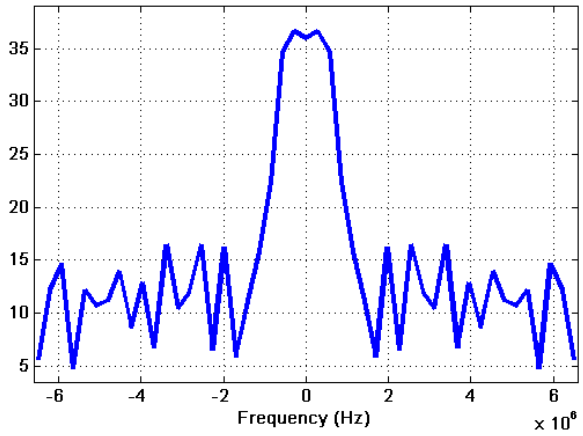


Fig. 2.25: Frequency response of a 29-tap low pass filter with cut-off frequency of 620 kHz

2.5.3. CXF2 modelling

The CXF2 block is a 127-order complex filter with 128 complex coefficients. It is a FIR architecture and the coefficients are generated with Matlab with the parameters as the order, the filter type, the sampling frequency, the cut-off frequencies and the passband and stopband weights setting. The modelling of this block is like the principle of the above LPF block but with complex input, output and coefficients. Since the FIR filter contains only a sum of the different input samples ($x[n], x[n-1], x[n-2], ...$), we just have to model this block with two real value inputs inI and inQ to represent the real and the imaginary parts of the complex value input (which is also the ADC output), multiply with the complex coefficients and then output the real part and the image part of this result separately as outI and outQ. The SC-AMS code of this modelling can be modified from the LPF and is shown on the next page.

```
……
sca_util::sca_vector< double > inI;
sca_util::sca_vector< double > inQ;
void processing()
{  ……
 std::complex<double> yout, sout;
 yout = std::complex<double> (0.0,0.0);
 sout = std::complex<double> (0.0,0.0);

 for( int i = N - 1; i > 0; i --)
 {  x(i) = x(i-1);
   yout += c(i)x(i);  }                          // complex value operation

 x(0) = std::complex<double> (inI.read(), inQ.read());
 sout = yout + c(0)x(0);                        // complex value operation

 outI.write(sout.real());
 outQ.write(sout.imag());
 ……}
……
```

The transfer function of this filter is shown in Figure 2.26. And the output spectrum of this filter is shown in Figure 2.27, where the out-of-band noise of the ΣΔ modulator output is reduced.

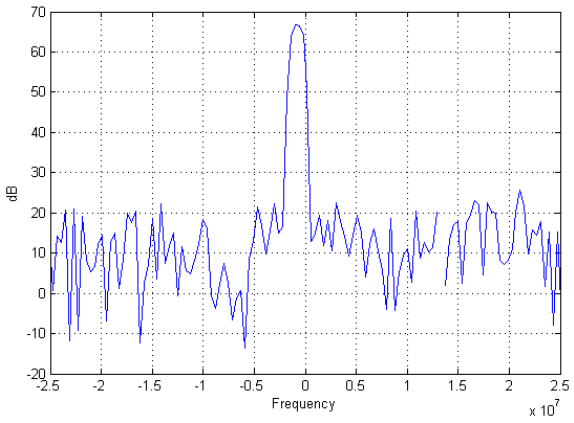


Fig. 2.26: Transfer function of the CXF2

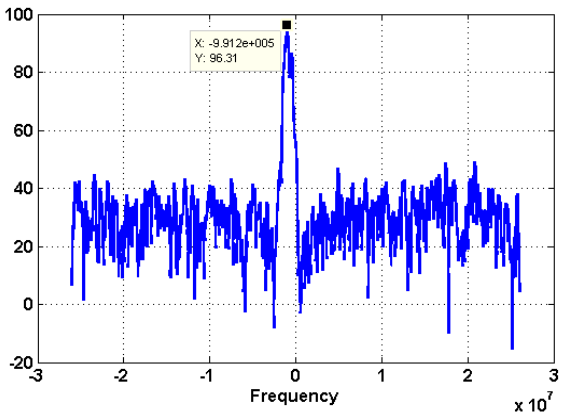
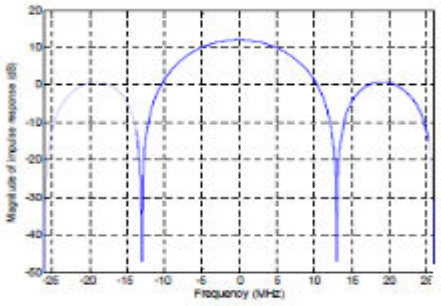


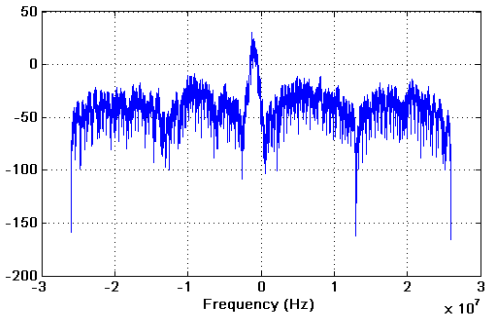
Fig. 2.27: CXF2 output spectrum

2.5.4. 4-tap filters

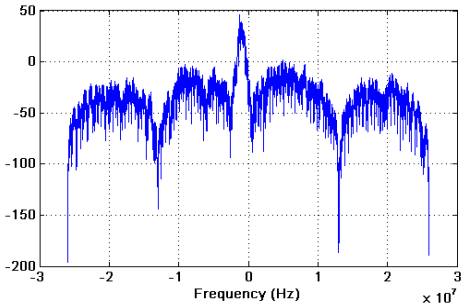
The 4-tap filter is a FIR filter with 4 coefficients. It is used to attenuate the noise which will be aliased in the useful frequency band by the down-sampling of 4. This filter has simple structure, the modelling process, which is with the same principle as LPF block, is ignored. The transfer function is given in Figure 2.28 (a) and the output spectrums of the two steps filtering with this filter are shown in Figures 2.28 (b) and 2.28 (c) respectively.



(a)



(b)



(c)

Fig. 2.28: (a) Transfer function of the 4-tap filter; (b) Output spectrum of the first 4-tap filter; (c) Output spectrum of the second 4-tap filter

2.5.5. Down-sampling of 4 and frequency shifting

The down-sampling of 4 block reduces the sampling frequency by taking 1 sample from each 4. After that, a 1 MHz frequency shifting is needed here to shift the signal from the low-IF frequency of $-1\,MHz$ to DC with a multiplication of the term $e^{-j\omega_{IF}t}$. The output spectrums of these two steps are shown in Figures 2.29 and 2.30 respectively.

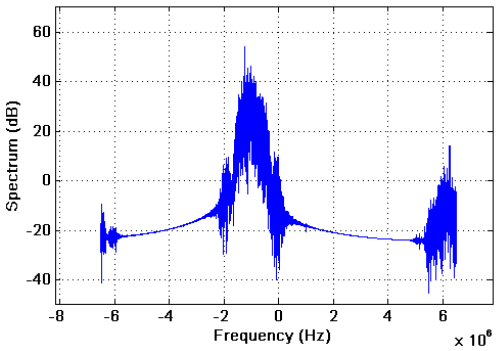


Fig. 2.29: Down-sampling of 4 output spectrum

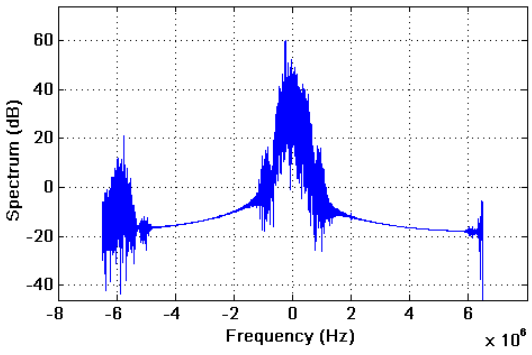


Fig. 2.30: Frequency shifting output spectrum

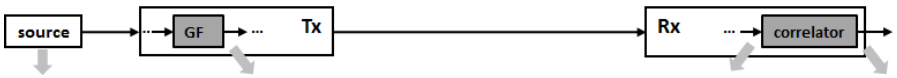
2.6. Transceiver functional model simulation

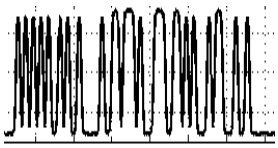
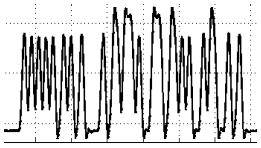
The first ideal version of the transceiver platform is functional. Before the refinement with RF/analog parameters, the simulation time has to be estimated. The first simulation takes

one BLE packet of 64 bits as the transceiver system input, and gives simulation results as the received bits and the simulation time.

The source packet contains only three parts as an 8 bits preamble, a 32 bits AA and a 24 bits PDU with values as shown in Table 2.1.

Table 2.1: Ideal transceiver simulation results



System input	GF output	Correlator input	Correlator output
1 BLE packet: **Preamble "0x55":** 0101 0101 (L) **AA "0x71764129":** 0111 0001 0111 0110 0100 0001 0010 1001 (L) **PDU "0x123456":** 0001 0010 0011 0100 0101 0110 (L)			**Received packet:** **Preamble "0x55":** 0101 0101 (L) **AA "0x71764129":** 0111 0001 0111 0110 0100 0001 0010 1001 (L) **PDU "0x123456":** 0001 0010 0011 0100 0101 0110 (L)
Simulation time: 18.71 seconds (64 bits) => 0.29s per bit			

From this result, it can be shown that the system works correctly. It can give in the correlator output the same packet sent in the source with a time delay. However, the simulation time is too long (about 19s for 64 bits) for just one BLE packet.

The later work concerns the transceiver system refinement and the co-simulation with the higher system where the exchanged information is in the form of packet or even command for the objective of being able to analyze the transceiver BER and the energy consumption of certain BLE use case. This is huge data volume for the transceiver which contains RF blocks. A slow transceiver model makes the global simulation impossible. Other techniques are needed to optimize the simulation time.

3. Simulation time optimization

To be able to simulate a large number of packets (e.g. 1000 packets), we use the baseband equivalent (BBE) method explained in chapter 1 [12]. The transceiver model is modified as a BBE one with $fs = 52$ MHz for the RF part which is the same simulation frequency as for the ΣΔ ADC in the receiver. As the spectrum out of the band of simulation ($\pm\frac{f_s}{2}$) are the repeats of the in-band spectrum, we will find the signal spectrum around ± 10 MHz (instead of ± 2.402GHz). The signal spectrum is repeated every 52 MHz, so we can find it around ± 10 MHz by using $f_c' = f_c - f_s \times \text{ceil}\left(\frac{f_c}{f_s}\right)$. This process is explained as shown in Figure 2.31.

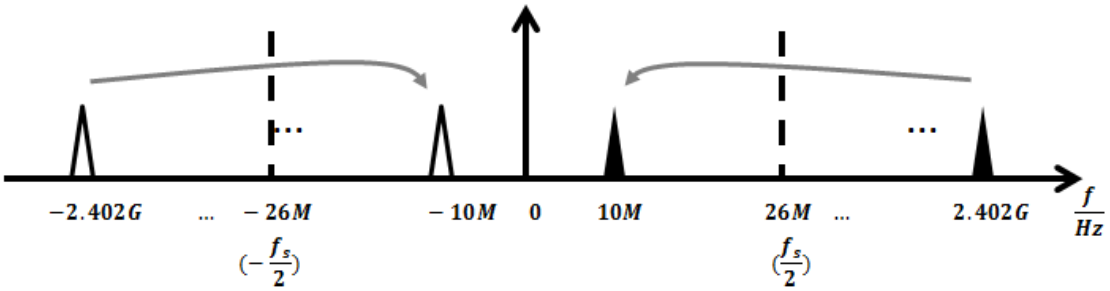


Fig. 2.31: Baseband equivalent simulation with simulation frequency as 52 MHz

The transceiver simulation platform is reconfigured as shown in Figure 2.32 with the new simulation frequency for the RF analog part as 52 MHz instead of 10.816 GHz. This simulation modification will tremendously reduce the simulation time of the whole system.

In the transmitter, the simulation frequency is raised from 13 MHz to 52 MHz from the LPF input, and is maintained during the entire RF analog process. The ΣΔ ADC working frequency of 52 MHz is the lowest value that can be used as the simulation frequency for this platform to maintain the system SNR performance at the output of the ADC. This is confirmed with the ADC output SNR.

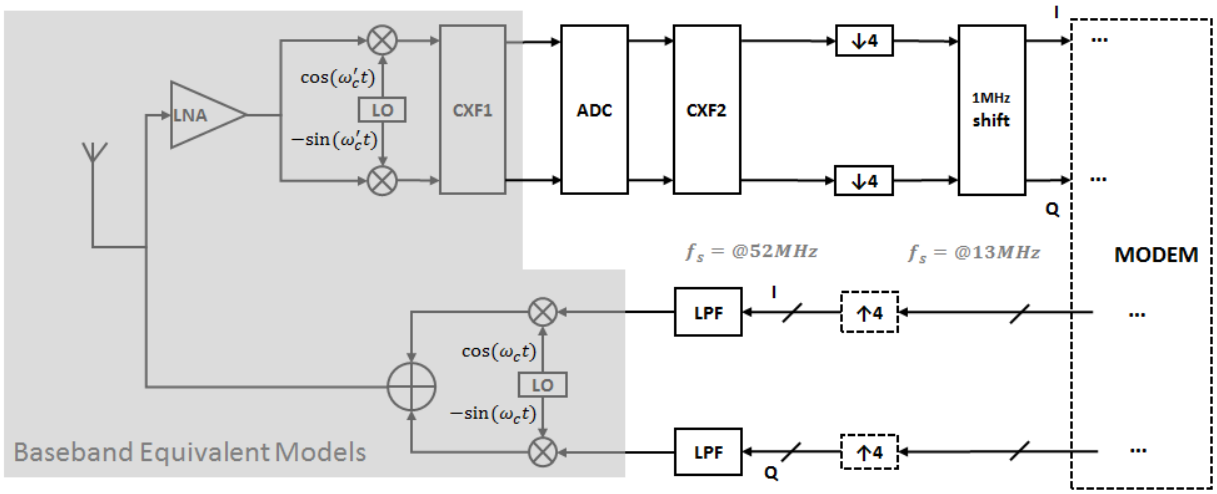


Fig. 2.32: Simulation frequency reconfigured transceiver simulation platform

This transceiver model is re-simulated with the same BLE packet. Since the ADC output SNR doesn't change, the transceiver works correctly as in the first simulation, but the simulation time is greatly reduced (cf. section 5, Table 2.11)

4. Rx front-end refinement with RF specifications

Since the Rx performance evaluation is the objective of this work, the RF specifications including the input/output impedance, the linear gain, IIP3, NF and power consumption should be considered in the Rx modelling. In particularly, the refined models need to be

verified in the baseband equivalent situation. In this section we will talk briefly about a C++ data type used in this work to enable the RF non-linear effects representation in the BBE (BaseBand Equivalent model) simulation. Then the refinements of each block are explained with the simulation results of verification. Finally the refined transceiver system model is estimated by a BER simulation.

The pair of LO carriers, which mix with the LNA output signal, comes from a frequency synthesizer which will be talked about in the end of this section. As described in section 2.4.3 in chapter 1, we introduce, in BBE simulations, a C++ data type of $BB\ (DC, I1, I2, I3, Q1, Q2, Q3)$ to represent a signal as the three harmonics and DC value to realize the representation of RF non-linear effects i.e. the gain-compression, the inter-modulation, etc. E.g. a BB type signal $x(DC, I1, I2, I3, Q1, Q2, Q3)$ [28] equals to a time domain signal $I(t)$ as

$$I(t) = DC + I_1 \cos(\omega_c t) + I_2 \cos(2\omega_c t) + I_3 \cos(3\omega_c t) + Q_1 sin(\omega_c t) + Q_2 sin(2\omega_c t) + Q_3 sin(3\omega_c t)$$

where t is the instantaneous virtual simulation time. This equation can be used to convert BB type to the traditional C++ data type i.e. *double*. The mathematical operations of signals with this type are also defined in the BB class.

4.1. RF channel model

The RF channel in this simulation is modelled as the input signal with power limited by a signal strength controller "ss" which converts the desired signal strength in dBm to an attenuate factor "a" as shown in Figure 2.33. In the Rx chain performance analysis, we use a single tone as the Rx input which has a fixed average power to measure the performance like SNR. It is easy to control the signal power with this method.

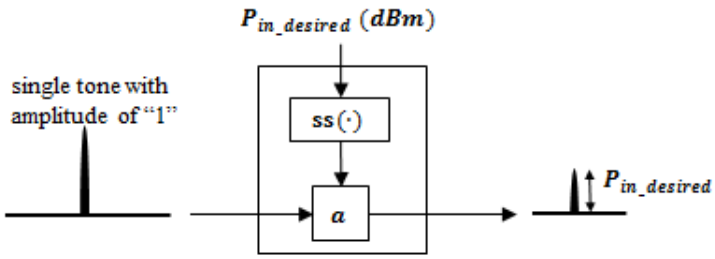


Fig. 2.33: Channel model

4.2. Receiver analog block refinement method

To model any analog/RF blocks, we start with the same description: ideal functional behavior. Then, we have to add for each block only the necessary extracted low-level specifications (non-ideality, some other low-level characteristics are neglected) in order to refine the model and perform accurate simulations.

In the receiver, the LO non-ideality will be abstracted as a phase noise which is a different analog specification in comparison with the previous ones, so it will be talked later. The rest main analog blocks to be refined are LNA, Mixer and CXF1.

In conclusion, each of the ideal functional models of these blocks is modelled as certain operation as shown in Equation 15. LNA output is a direct equation of the input. The Mixer output is a multiplication of the two inputs, and the filter output is the function F_{CXF1} operated the input. The output of each block can be considered as a function of the input like $out = F(in)$ in general.

$$\begin{bmatrix} LNA_{out} \\ Mixer_{out} \\ CXF1_{out} \end{bmatrix} = \begin{bmatrix} in_{LNA} \\ in_{Mixer} \times in_{LO} \\ F_{CXF1}(in_{CXF1}) \end{bmatrix} \quad \text{(Eq.15)}$$

Let us considering the signal amplitude as voltage (rather than current (A) or power (dBm)). The total noise added by a block can be modelled as a referred-to-input thermal noise density δ due to the input residence R as $\delta = \sqrt{4(10^{NF/10} - 1)KTR\frac{f_s}{2}}$. The non-linearity can be modelled with considering only the fundamental wave and the 3rd-order harmonic as the function $F_r(x) = \alpha_1 x - \alpha_3 x^3$ where x is the original input (plus the referred-to-input noise density as $in + \delta$). α_1 is the linear gain of the block and α_3 is the 3rd harmonic coefficient calculated by using the relationship of $A_{IIP3} = \sqrt{\frac{4}{3}\left|\frac{\alpha_1}{\alpha_3}\right|}$ which was introduced in chapter 1.

For each block, described by Eq. 15, the model is refined by the function $F_r(x)$ in order to take into account its non-linearity (IP3). This refinement flow is shown in Figure 2.34.

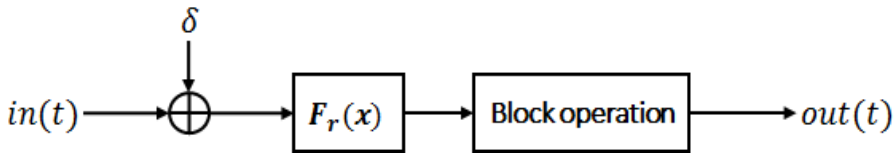


Fig. 2.34: RF/analog block refinement

4.3. Refinements of Rx analog/RF blocks

The analog/RF blocks in the receiver are refined using the method described in the last subsection. The refined models are validated in this subsection. The verification simulations take the sinusoidal waves as inputs which are given in form of average power value in dBm. The real circuit measured RF specifications are taken as the input of the RF effects generation and is verified by comparing the corresponding block model simulation with the measured results from Riviera Waves Company circuits.

4.3.1. LNA refined model verification

For the receiver, the specification of the LNA are : linear gain of 20 dB, $NF\ of$ 2.9 dB and $IIP3$ of $-15\ dBm$ (values given by Riviera Waves Company).

LNA model noise figure verification

LNA NF simulation is represented in Figure 2.35 and the simulated results are given in Table 2.2. We have given a value of 2.9dB for the NF as a generic parameter of the high-level model. Then, we realize a simulation and we calculate the NF, by computing the SNR_{out} and the SNR_{in} in the spectrum of the input and output signal (cf. Fig. 2.35).

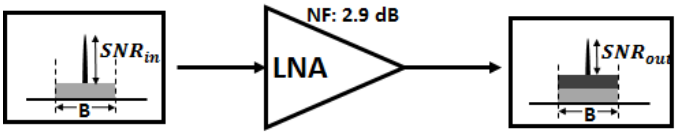


Fig. 2.35: LNA noise figure simulation

Table 2.2: LNA noise figure verification

NF: 2.9dB	$\boldsymbol{SNR_{in}}$	$\boldsymbol{SNR_{out}}$	$\boldsymbol{NF_{simu}}$
$\boldsymbol{S_{in}}$: $-80dBm$	15dB	12.074dB	**2.926dB**
$\boldsymbol{N_{in}}$: $-95dBm$			

The LNA simulated NF value is close to the input NF specification (generic value). So, we can consider that the NF modelling is validated.

LNA model Linear Gain and IIP3 verification

Linear gain is calculated by the difference between the output and input power. The result is given in Table 2.3.

Table 2.3: LNA linear gain verification

Gain : 20dB	$\boldsymbol{S_{out\_simu}}$	$\boldsymbol{Gain_{simu}}$
$\boldsymbol{S_{in}}$: $-80dBm$	−60,107dBm	**19.893dB**

Then, we have estimated the IP3 of the LNA, with a generic value of -15dBm. The two adjacent signals generated at the LNA input are:

$$s_1 = 4.4721\mathrm{e}^{-6}\cos(2\pi f_1 \mathrm{t})$$
$$s_2 = 4.4721\mathrm{e}^{-6}\cos(2\pi f_2 \mathrm{t})$$

where $f_1 = 2401.875MHz$ and $f_2 = 2402.125MHz$, the amplitude of $4.4721\mathrm{e}^{-6}$V corresponds to a power of -80dBm.

The simulated output spectrum is shown below in Figure 2.36. In the IIP3 test, the input signals should be small enough to avoid being obviously compressed at the LNA output. This can be ensured by comparing with the linear output component which is $S_{in} \times$ LNA_linear_gain (the grey curb in Fig. 2.36). In this figure, LNA output has the same power as the linear output component, which is the appropriate situation to measure the IIP3.

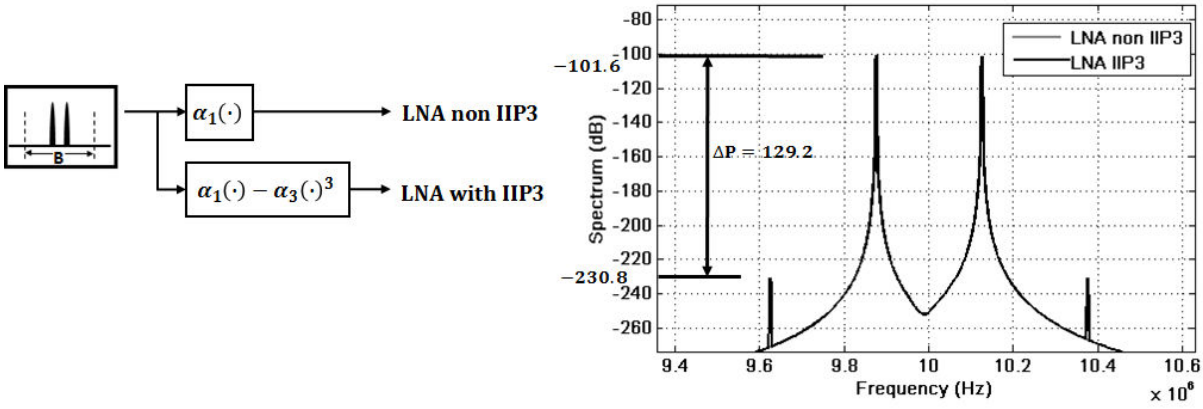


Fig. 2.36: LNA IIP3 verification

The IP3 is estimated using equations 16 and 17.

$$\text{Pin} = \frac{\text{Ain}^2}{2},\ \text{Pin}_{\text{dBm}} = 10\log10(Pin \times 1000) = -80\ \text{dBm} \qquad \text{(Eq.16)}$$

$$\text{IIP3}_{\text{dBm}} = \text{Pin}_{\text{dBm}} + \frac{\Delta\text{P}}{2} = -80 + \frac{129.2}{2} = -15.4\ \text{dBm} \qquad \text{(Eq.17)}$$

We have verified that the estimated gain and the IP3, calculated from the input and output signal of the LNA are very close to the generic values included in the high-level model. So, we can consider that the gain and IP3 modelling is validated.

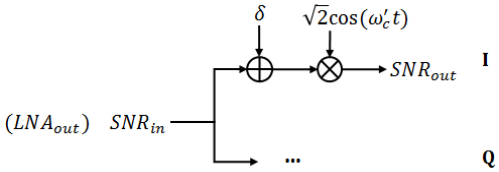
4.3.2. Mixer refined model verification

In the receiver specification, the RF mixer is described with the linear gain of $3\ dB$, NF of $13.2\ dB$ and $IIP3$ of $-5\ dBvrms$. Each specification (i.e. each verification) is observed at any of the two branches (I or Q).

Mixer noise figure model verification

Mixer has two inputs in each of the two branches (I and Q) which are the signal input and the RF carrier. The RF carrier wave is an ideal sinusoidal wave in this simulation. $LNA_{out} = \cos(\omega_c t)$ with an average power of 0.5 Watt, which indicates a contribution of $-6\ dB$ for the gain of the mixer, and this is not correct because LO isn't supposed to contribute any extra gain in the mixer verification. So we add a factor of $\sqrt{2}$ to avoid contributing any extra gain on the side of LO as $\sqrt{2}\cos(\omega_c' t)$ and $\sqrt{2}\sin(-\omega_c' t)$ for the branch I and the branch Q respectively where $\omega_c' = \omega_c + 2\pi * 1MHz$ (f_{FI}=-1MHz).

Table 2.4: Mixer noise figure verification



NF_{dB}: 13.2 dB	SNR_{in}	SNR_{out}	NF_{simu}
S_{in}: $-80dBm$	$15dB$	$1.772dB$	**13.228dB**
N_{in}: $-95dBm$			

NF simulation is realized the same way as in LNA NF verification and simulation result is shown in Table 2.4. One more time, the block model measured NF value is close to the input NF specification. So, we are considering that as NF model is validated.

Mixer linear gain and IIP3 model verification

The linear gain verification is directly given in Table 2.5.

Table 2.5: Mixer linear gain verification

Gain: $3dB$	S_{out_simu}	$Gain_{simu}$
S_{in}: -80dBm	-76.983dBm	**3.017 dB**

The channel around 2.402 GHz is chosen as an example for the IIP3 simulation. The two adjacent signals generated at the Mixer input are:

$$s_1 = 4.5e^{-4}\cos(2\pi f_1 t)$$

$$s_2 = 4.5e^{-4}\cos(2\pi f_2 t)$$

where $f_1 = 2401.875MHz$ and $f_2 = 2402.125MHz$. These two signals are fed into both of the I and Q branches like an LNA output, where LNA is just a register.

The simulated output spectrum is shown below on the right side of Figure 2.37. In this figure, the Mixer output ("MixI_IIP3" in Figure 2.37) has the same power as the linear output component ("MixI_non_IIP3" in Fig. 2.37, i.e. -40dBm or $4.5e^{-4}$V), which is the appropriate situation to measure the IIP3.

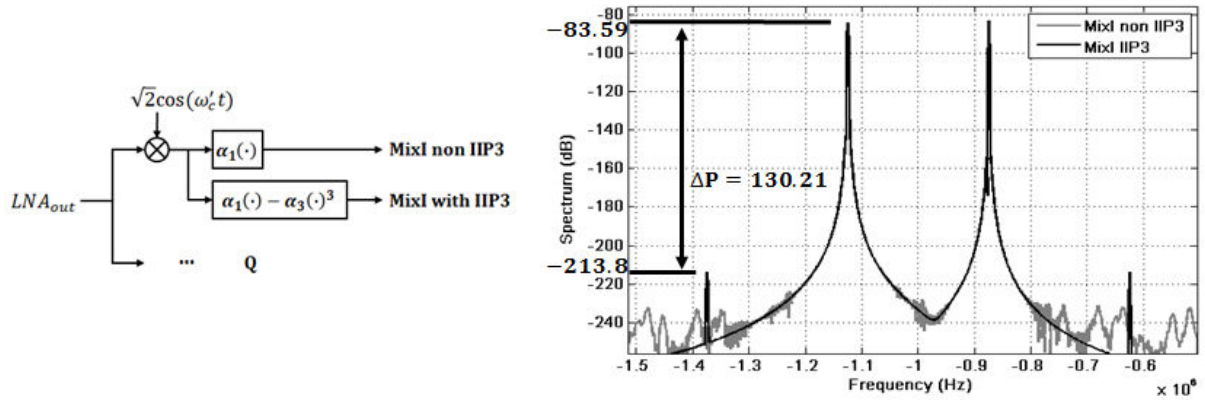


Fig. 2.37: Mixer IIP3 verification

The IP3 is estimated by the following equations:

$\mathrm{Pin} = \frac{\mathrm{Ain}^2}{2} = -40\ (\mathrm{dBm})$,

$\mathrm{IIP3_{dBm}} = \mathrm{Pin_{dBm}} + \frac{\Delta \mathrm{P}}{2} = -40\ (\mathrm{dBm}) + \frac{130.21}{2} = 25.105\ (\mathrm{dBm}) = -4.895\ (dBvrms)$.

These results are validating our model.

4.3.3. Analog complex filter refined model verification

The CXF1 specifications, given by Riviera Waves Company, are respectively: linear gain of $26\ dB$, NF of $13.2\ dB$ and $IIP3$ of $-11\ dBmAp$ (equals to $-41\ dBm$).

CXF1 noise figure model verification

CXF1 NF simulated results are summarized in Table 2.6. The validation methodology is the same: we put as generic parameters the NF value. Then, we realize the simulation and we calculate the input and the output SNR (using Matlab) and finally, we estimate the NF.

Table 2.6: CXF Noise Figure verification

NF: $13.2dB$	$\boldsymbol{SNR_{in}}$	$\boldsymbol{SNR_{out}}$	$\boldsymbol{NF_{simu}}$
$\boldsymbol{P_{in}}: -80dBm$	$14.965dB$	1.682dB	**13.28dB**
$\boldsymbol{N_{in}}: -95dBm$			

CXF1 linear gain and IP3 model verification

The linear gain verification is directly given in Table 2.7.

Table 2.7: CXF1 linear gain verification

$\boldsymbol{Gain_{dB}}$: $26\ dB$	$\boldsymbol{S_{out\_simu}}$	$\boldsymbol{Gain_{simu}}$
$\boldsymbol{S_{in}}: -80dBm$	-53.79 dBm	**26.21 dB**

To verify the IP3 value, we consider that the interested band of the CXF1 filter is from -1.5 MHz to -0.5 MHz. The two close tones are respectively $inputI = \mathrm{A_{in}}\cos(2\pi f_1 \mathrm{t})$ and $inputQ = \mathrm{A_{in}}\cos(2\pi f_2 \mathrm{t})$, where $f_1 = -0.875\mathrm{MHz}$, $f_2 = -1.125\mathrm{MHz}$ and $A_{in} = 1.00119e^{-7}\ V$ corresponding to an input signal power of -113dBm. The simulated result (output spectrum) is shown in Figure 2.38. This result gives: ΔP =144.1dB. So, the IIP3 can be estimated by the following equation:

$$\mathrm{IIP3_{dBm}} = \mathrm{Pin_{dBm}} + \frac{\Delta \mathrm{P}}{2} = -113 + \frac{144.1}{2} = -40.95\ (\mathrm{dBm}) \qquad \text{(Eq. 18)}$$

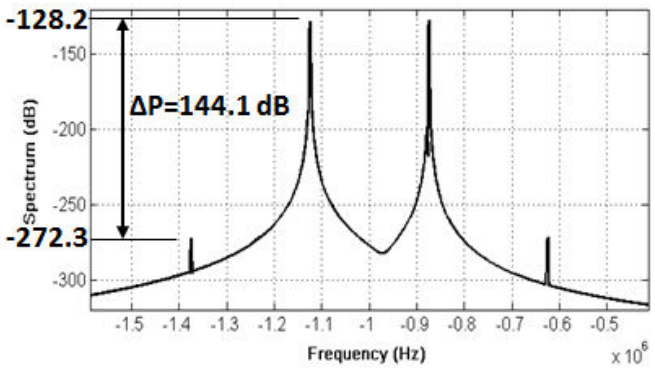


Fig. 2.38: Rx CXF SC-AMS model IIP3 simulation output spectrum

4.3.4. ADC modelling, verification and characterization

The high level modelling of the ADC can be considered as the input added with the overall measured block noise. In the receiver specification, the ADC is described with NF equals to 39.2 dB.

ΣΔ-ADC noise figure verification

We set $NF_{ADCdB} = 39.2dB$. Since the NF of the ADC is high, we put the input signals with a signal power higher than before as $-47\ dBm$. With the 0.83 dB scallop loss of the Blackman-Harris window [29], the input signal power calculated from the simulation is $P_{sig}\_w\_th = -47.83dBm$. The total theoretical noise is $P_n\_w\_th = -88.795\ dBm$, corresponding to an input $SNR_i\_th = 40.965$ dB. The simulated output gives a value of SNR_o_th $= SNR_i$_th $- NF_{ADCdB} = 1.765$ dB.

ADC NF simulated result is shown in Table 2.8, which is close to the NF specification given as a generic parameter of the model.

Table 2.8: ADC noise figure verification

NF_{ADCdB}: 39.2 dB	SNR_{in}	SNR_{o_simu}	NF_{simu}
P_{in_th}: $-47dBm$	41.8dB	1.761 dB	**40.04 dB**
N_{in_th}: $-88.8dBm$			

4.4. Cascaded specifications verification

With the RF analog models verified before in the last section, we are able to verify the Rx front-end chain parameters as chain NF and IIP3. In this simulation, the specifications are supposed as that the Rx mixer has a maximum gain mismatch of 6 dB which is due to the

impedance mismatch. This situation corresponds to the effect of setting the LNA linear voltage gain as $\frac{\text{Gain\_LNA\_v}}{2}$.

4.4.1. Rx front-end NF simulation

To verify the noise figure, the IIP3 parameter is set close to infinite as 10^5dBm or 10^5dBmAp for each block for not having the non-linearity effect in this simulation.

The parameters of Gain and NF of each Rx block are configured as in Table 2.9.

The Rx input signal power is set to $P_{sig} = -70\text{dBm}$ which is generated in "channel", and the antenna generates a thermal noise of the 50 Ohm equivalent resistance which is equivalent to $P_n = 4KTRf_s = 8.0078e^{-13}\ V^2 = -90.965$ dBm, which makes an input $SNR_i = P_{sig} - P_n = 20.96$ dB.

The theoretical cascaded NF value at the output of each block is calculated as shown in Table 2.9 ($NF\_accu\_th\ (dB)$).

The simulation results are listed in the last two lines in Table 2.9. The simulated noise figure values (NF_casc _simu (dB)) values are close to the theoretical values which are located above.

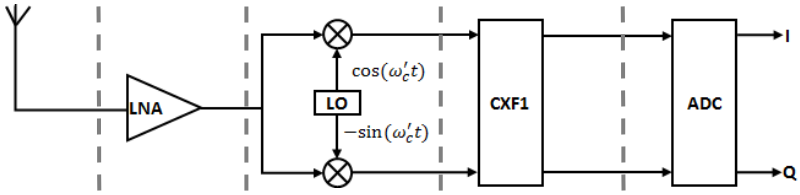


Table 2.9: Rx analog NF simulation results

	Channel	Antenna	LNA	Mixer	CXF1	ADC
P_{in} (dBm)	-70					
P_n (dBm)		-90.965				
Gain (dB)			20	3	26	
NF(dB)			2.9	13.2	15.1	39.2
IIP3			1e5 dBm	1e5 dBvrms	1e5 dBmAp	
NF_casc_th (dB)			2.9	4.4	5.3	5.8
NF_casc_simu (dB)			**2.88**	**4.37**	**5.22**	**5.73**

4.4.2. Rx front-end IP3 simulation

We configure firstly our system with the specifications of each block in Table 2.10. Then, we apply two tones as $inputI = A_{in}\cos(2\pi f_1 t)$ and $inputQ = A_{in}\cos(2\pi f_2 t)$ at the

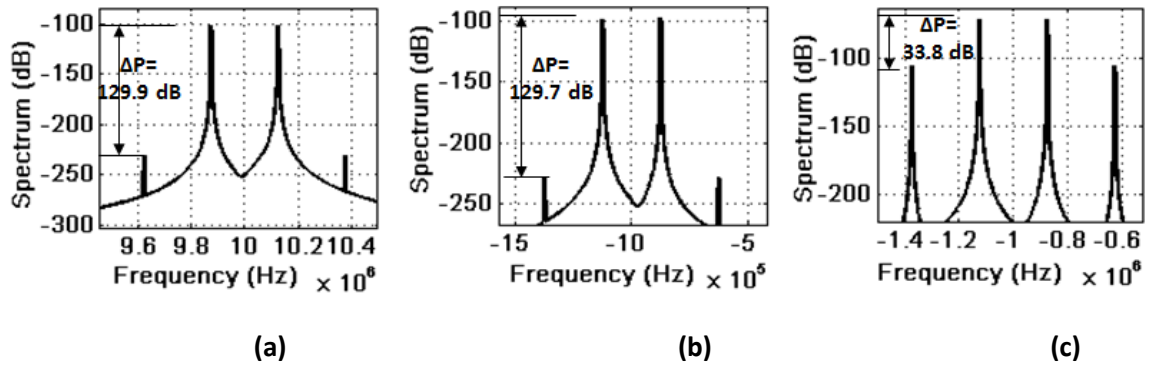
input of this system, where f1 = 2401.875 Hz, f2 = 2402.125 Hz and $A_{in} = 4.4721e^{-6}$ V which corresponds to an input signal power of −80 dBm.

Cascaded 3rd intercept point is estimated with this equation $\frac{1}{A_{IP3}^2} = \frac{1}{A_{IP3,1}^2} + \frac{A_{V1}^2}{A_{IP3,2}^2} + \frac{A_{V1}^2 A_{V2}^2}{A_{IP3,3}^2} + \cdots$, which is used to calculate the theoretical cascaded IIP3 values in Table 2.10.

By running the simulation, we obtain the signal spectrum at the output of each stage in this system as shown in Figure 2.39 and the results in the last line of Table 2.10.

Table 2.10: Rx chain cascaded IIP3 verification

Pin: -80dBm	**LNA**	**Mixer**	**CXF**
Gain (dB)	20	6	20
IIP3_spec (dBm)	-15	25	-41
IIP3_th_casc (dBm)	-15	-15.04	-64
IIP3_th_simu (dBm)	**-15.1**	**-15.325**	**-63.135**



(a) (b) (c)

Fig.2. 39: Cascaded IP3 results at the output of (a): LNA, (b): Mixer, and (c): CXF

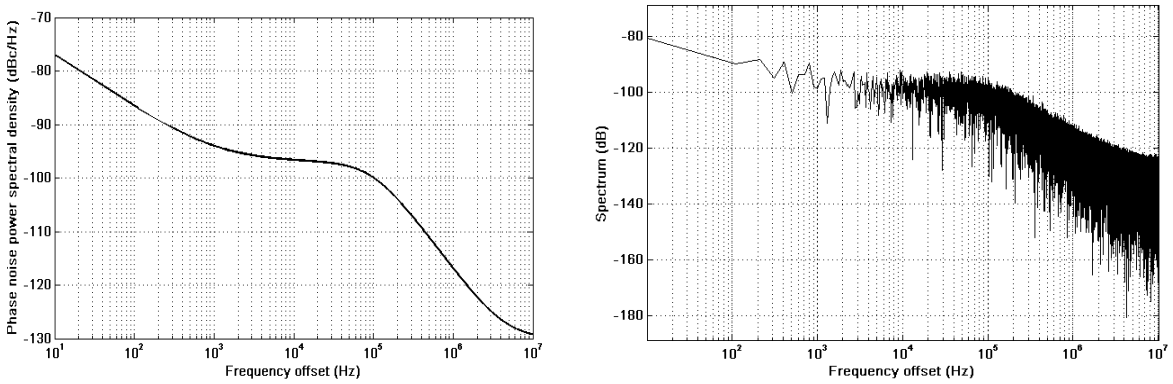
4.5. PLL phase noise

The model of the phase noise is based on the Figure 1.13 shown chapter 1 [6]. It consists on sub-blocks generating $1/f$ component, $1/f^2$ component and the 2 noise floors. The Equation 19 reminds this model.

$$L(f_\Delta) = \frac{B_{PLL}^2 L_0}{B_{PLL}^2 + f_\Delta^2}\left(1 + \frac{f_{corner}}{f_\Delta}\right) + L_{floor} \qquad \text{(Eq.19)}$$

It is defined by the PLL bandwidth B_{PLL}, the corner frequency f_{corner}, and the in-band and out-band noise floors L_0 and L_{floor}. Our PLL PN specification is equal to $-125\frac{dBc}{Hz}$@3 MHz. We choose the phase noise model parameters to adapt this specification as: $B_{PLL} = 100$ KHz, $L_0 = -97$ dBc/Hz, $L_{floor} = -130$ dBc/Hz, $f_{corner} = 1$ KHz. The

phase noise profile described by this definition is shown in Figure 2.40 (a) and the corresponding phase noise spectrum is shown in Figure 2.40 (b).



(a) (b)

Fig. 2.40: (a) Phase noise profile; (b) Phase noise measurement of a circuit

5. Transceiver simulation and BER estimation

To estimate the BER of the transceiver, the Rx chain is configured as Table 2.11. The simulation is run by sending 1000 BLE packets in the data source with the packet interval between packets of one packet length (cf. Fig. 2.41). Each packet contains 64 bits. The total virtual time to simulate is 128000 µs.

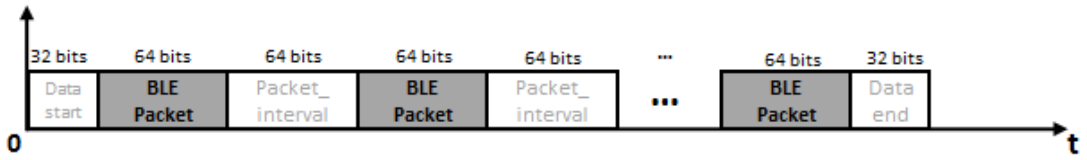


Fig. 2.41: Data source for the Cascaded Rx front-end verification

With the RF chain configured, we simulate the receiver with sending it 1000 packets of 64 bits. The input signals are with powers from $-93dBm$ to $-91dBm$ and the receiver noise floor is $-114dBm$. The obtained BER performance and Rx sensitivity is shown in Fig. 2.42. From this simulation, the Rx sensitivity is $-93.3\ dBm$ which corresponds to a BER of 0.1%. Another simulation result is given as the simulation time which is shown in the last line in the Table 2.11. It takes only 470s for a simulation of 2000 packets (1000 data packets and 1000 empty interval packets). This duration is acceptable and makes it possible for a future global BLE system simulation.

In comparison with the Table 2.1, where the simulation time was about 0.3s/bit, the speed-up is equal to 80. The sensibility of the receiver is about -93.3dBm, which correspond to a SNR equals to 20.7dB (Floor noise=-114dBm).

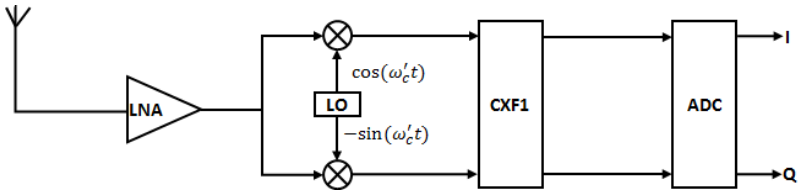


Table 2.11: Transceiver BER analysis configuration

	Channel	Antenna	LNA	Mixer	CXF1	ADC
P_{in} (dBm)	From $-$ 96dBm to $-$ 91 dBm					
Noise floor (dBm)		-114				
Gain (dB)			20	3	26	
NF(dB)			2.9	13.2	15.1	39.2
IIP3			-15 dBm	-5 dBvrms	-11 dBmAp	
Phase noise	-125dB/Hz@3MHz					
Sensitivity (dBm)	**-93.3dBm**					
Simulation time	**469.62 s ➔ 3.7ms/bit**					

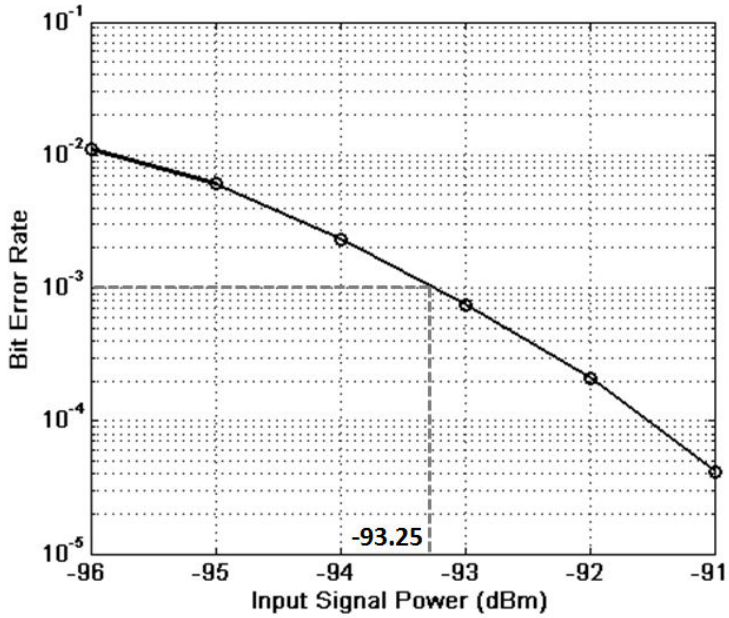


Fig. 2.42: BER and Rx sensitivity simulation results

6. Conclusion

In this chapter we have introduced the system-level modelling in SCAMS for a BLE transceiver. The first version was an ideal functional model which was verified by running a simulation which took a single BLE packet as input and gave the correct received one at the

transceiver output. The problem of this model is the prohibitive simulation time (i.e. 0.3s/bit) which will make it difficult for the future BLE system simulation to be carried out.

BBE method was applied to reduce the simulation time. The optimized transceiver model was re-simulated with in the same way as the precious model. It worked correctly and the simulation time was dramatically reduced (i.e. 3.7ms/bit).

The analog RF part of the receiver chain was refined with the main RF specifications and was verified by the analysis of the analog RF effects from a series of block-level and Rx front-end simulations. The receiver performance was finally estimated as BER by running the refined model with 1000 BLE packets.

In the next chapter, RF blocks will be built with configurable models for easy reusing for an efficient BLE system simulation, or for future applications like performance analysis of a modified Rx chain (block specifications or chain architecture).

The configurable transceiver model will be integrated into the SCTLM BLE digital platform with an interface which will connect the two domains with the RF control signals from the digital and the data conversion.

Chapter III – System modelling and simulation

1. Introduction

In the last chapter, the BLE transceiver was modelled and verified at system-level. The simulation results presented the possibility of a global system simulation with this model including the analog/RF characteristics.

In this chapter, this model will be applied into a global BLE communication network simulation aimed at the energy consumption estimation of the RF part during a certain BLE use case. In the global simulation, transceiver block models are needed to be easily modified to adapt different requirements. In the case of the modification in the transceiver design, we won't be willing to re-model a certain block each time and replace it with another design, or even we change the system architecture. As the blocks of different groups have common features, we can establish a common model for each group, and each time it only needs to define a certain block by reusing the same model but with different specifications.

In the other side, the whole digital system is modelled in SCTLM with the lowest system of LL. For connecting the transceiver to LL, it needs to replace the original "channel" model with the SCAMS transceiver model, and to create the control link between the digital domain and the analog domain (cf. Figure 3.1). The control signals includes the transceiver part enabling signals with which we are able to estimate the energy consumption of each part of the transceiver. It is very useful to understand the relationship between the energy consumption and the performance of a transceiver during a given application in the new design.

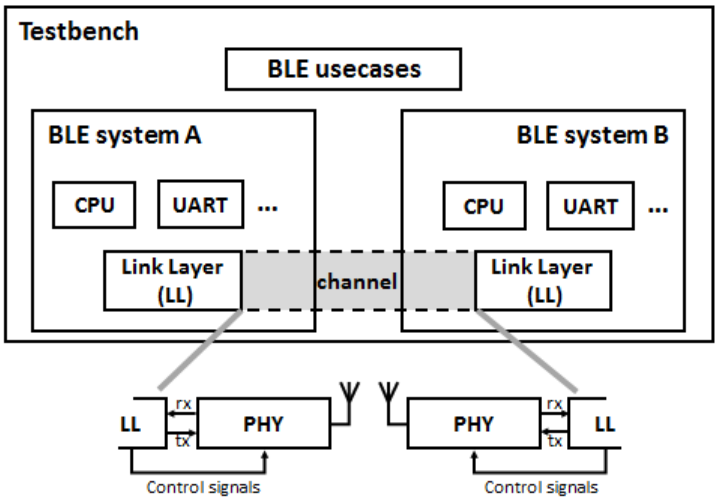


Fig. 3.1: Transceiver integration into SCTLM described high-level test bench

2. Parametrized receiver blocks

2.1. Introduction

In the global simulation of the system, we create a top netlist to establish the components (i.e. building blocks or instances) which construct the system. For the intent of briefness, the script of this netlist shouldn't contain the details of all of the instances. If the amount of these components is large and we need to modify all models often in the system simulation, we need to modify all the files of models and re-test them all the time, which makes the global simulation time-consuming and error-prone. In this work, to be able to compare the energy consumptions and the performances of the system with different blocks or even with different architectures, we will have to modify the netlist of the receiver frequently. So the receiver netlist should be based on the model with higher level of abstraction.

In the receiver front-end, there are certain blocks having common characteristics, e.g. the RF blocks can be described by the same types of RF characteristics, or the filtering can be represented by the transfer function in a unified form. Building a common basic model, which contains the possible characteristics to represent a group of blocks, will be convenient for a netlist creation for the global simulation.

2.2. Parametrized LNA

The LNA is the most basic RF block which contains all the important RF characteristics. They can be commonly added as we have talked in the last chapter. The only thing to be unified here is the unit problem. In this work the input NF and IIP3 are unified in dBm, and the input linear gain is unified in dB, as depicted in Figure 3.2.

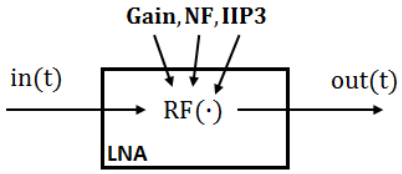


Fig. 3.2: Parameterized LNA model

2.3. Parametrized analog filters

CXF1 is a complex band-pass filter of the receiver. Its transfer function $H(s)$ contains also some complex coefficients. As the transfer function of analog blocks is not easy to create with existing SCAMS filter models, a complex filter can be considered as a corresponding Δf shifted low-pass filter, where Δf is the central frequency of the complex filter. CXF1 includes

all the extra steps in comparison with the analog LPF. So, CXF1 model will be the analog filtering common model with optional parameters.

The transfer function of analog filters can be written either in the form of zeros and poles (Eq.1), or in the form with the coefficients of different order harmonics (cf. Eq.2).

$$H(s) = k\frac{\prod_{i=0}^{M-1}(s-zero_i)}{\prod_{i=0}^{N-1}(s-pole_i)} \quad \text{(Eq. 1)}$$

$$H(s) = k\frac{\sum_{i=0}^{M-1} num_i \cdot s^i}{\sum_{i=0}^{N-1} den_i \cdot s^i} \quad \text{(Eq. 2)}$$

The equivalent model can be created to adapt both situations with an input indication. However, in this work, we have chosen to write it using Eq.2. The common model of the analog filtering is as shown in Figure 3.3.

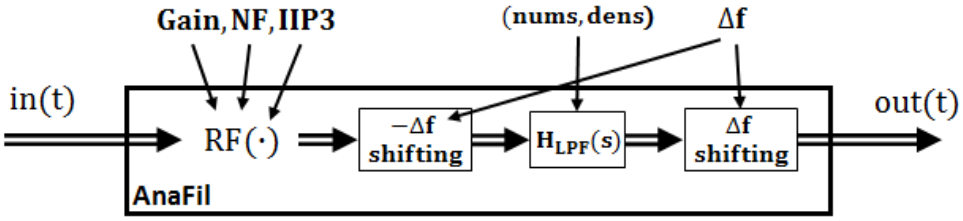


Fig. 3.3: Parameterized analog filtering model

In this model, the analog characteristics are added firstly like with the LNA. The second step is a common filtering which takes the shifted frequency and the LPF coefficients as generic parameters. As the input and output are both complex signals which are transmitted in quadrature paths (I and Q) in a complex filtering, the model is created with quadrature input and output. In case of building a real number filtering, the shifted frequency and the imaginary parts of the input and the output should be set to zero.

2.4. Digital filtering (ADC and D-filters)

Digital filters are not in the RF front-end of the modelled transceiver. However, the ΣΔ ADC can be described with the signal and noise transfer functions as STF(z) and NTF(z) respectively, and there is a lot of digital filtering blocks in the receiver digital signal process. So, it is necessary to create a common digital filtering model for this work.

The model to be built should be common that it can be used to describe all of the digital real and complex filtering. Based on the method of realizing digital filters (cf. last chapter), this common model (cf. Fig. 3.4) is built with quadrature inputs and outputs, and the filter coefficients are complex numbers. Considering the ADC noise contribution, this block takes NF as an optional generic parameter. We suppose that the quantification noise is out of the band of interest, and the total noise added by ADC is defined by NF. To realize the noise

shaping, we generate an internal noise filtered by NTF(z), where δ_q is the noise density of a theoretical quantification noise and it has no effect to the output SNR.

The transfer function of a digital filter can be written using two different types of equations as for the analog filtering. In this model, it is also written in the form with the coefficients of different order harmonics illustrated by Eq. 3.

$$H(z) = k\frac{\sum_{i=0}^{M-1} num_i \cdot z^{-i}}{\sum_{i=0}^{N-1} den_i \cdot z^{-i}} \qquad \text{(Eq. 3)}$$

The filter coefficients and input and output signals are also complex number. In case of ΣΔ ADC creation, there are complex coefficients for STF and NTF generation. In case of digital filtering, the NTF path should be initiated as zero by setting the filter order as zero. In case of LPF, the parameters including the Q data path are also set as complex number where the imaginary part is equal to zero.

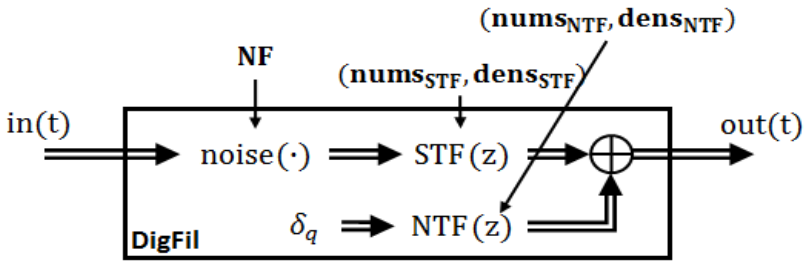


Fig. 3.4: Parameterized digital filtering model

Example : ADC and CXF2 model generation with "DigFil"

An example of ΣΔ ADC automatic model generation in the receiver netlist is given below. In the netlist, the ADC is created by calling "DigFil" function with the signal gain as "*stf_gain*", the coefficients of STF and NTF as "*sa,sb,na,nb*" and the filtering order as "3".

```
SC_MODULE(receiver){
  ……
 SC_CTOR (receiver) {  ……
   adc_sd=new DigFil("ADC",stf_gain,sa,sb,na,nb, 3);
   ……}
 ……}
```

The output spectrum is shown in Fig. 3.5, with an input signal of -45dBm. The ADC NF is verified and the result is given in Table 3.1. These results validate the simulation of the NF value (as well as the output spectrum of the ΣΔ ADC), so the automatic generation of the model, using "DigFil".

Table 3.1: ADC Noise Figure verification

$NF_{dB} = 13.2dB$	SNR_{in}	SNR_{out}	NF_{simu}
$P_{in}: -80Bm$	15dB	1.772dB	**13.228dB**
$N_{in}: -95dBm$			

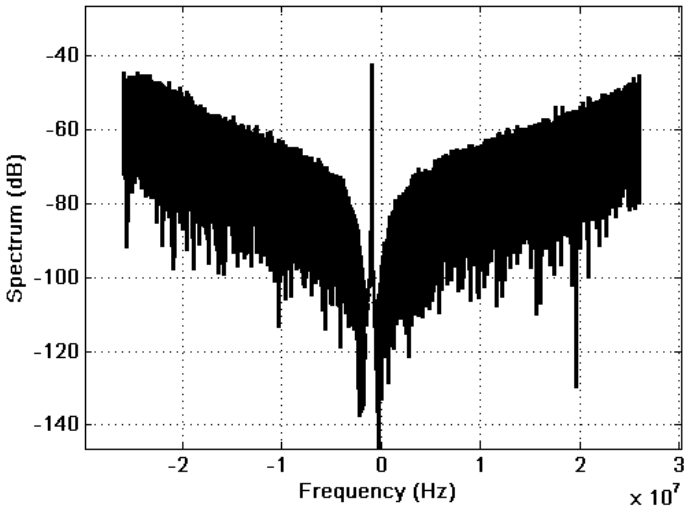


Fig. 3.5: ΣΔ ADC output spectrum by using "DigFil"

Another example of the automatic model generation is illustrated with a digital complex filter called "CXF2" in our receiver digital chain. In the receiver netlist, CXF2 is created by calling "DigFil" function with gain equals to "`1.0`" (or 0 dB), coefficients of STF and NTF equal to "`sa,sb,na,nb`" where "`na,nb`" are both zero. The filtering order is equal to "127". The simulated results of this block are identical as the original modelling, described in chapter 2, we will be not repeated here. We just give a part of the netlist below:

```
SC_MODULE(receiver){
 ……
 SC_CTOR (receiver) {
  ……
  cxf2=new DigFil("CXF2",1.0,sa,sb,na,nb, 127);
  ……}
 ……}
```

3. SCTLM platform at LL level and PHY interface

3.1. Introduction

The original platform of BLE network, developed bry Riviera Waves Company, was established to test some BLE applications. In this case, there is no PHY level because it is not needed for high level verifications. The lowest abstraction level of the BLE devices, in this platform, was LL. The LL is described as a series of "thread" which define the BLE LL performances according to the LL states specification presented in chapter 1.

The LL state machine is described by means of threads. In these LL threads, we are interested only in transactions where there are function calls towards "Air" model which indicate the transceiver activation or deactivation. We don't care about high level controlling or errors as they can't be seen by PHY. In the latter transceiver integration, this original

model will be almost maintained but with new created controlling and data interfaces. It is necessary to take a look at the data exchange in these threads to understand the possible controlling signals and data interface which can be used for the integration of the BLE transceiver model. Since the passive scan is the simplest state in LL, we explain passive scan thread in detail to have an idea of the LL model functionality in this platform. The other states are described with the same principle during the data exchange.

3.2. Passive scan thread

The passive scan thread is triggered by the passive scan event "*sc_core::sc_event pscan_evt*", then LL will open a scan window to wait for the advertising packet. This process is realized by using a loop which begins by calling *rx()* function in "Air" with specifying the scan window length "*winsize*" and wait until receiving a packet or until the end of the scan window. In the case of receiving a packet, *rx()* function returns the received packet and the rest of the window length for the next round of the loop. In the case of the end of the scan window or any interrupt, the thread is restarted by waiting for the passive scan event. The code below is the SC-TLM description of the passive scan mode with details of the key behavior while ignores the controlling details like the dealing with all kinds of abort, reset or frame high-level filtering, etc.

```
BLE_LL::BLE_LL(……)
{ ……
  struct ble_frame{ … };    // Definition of a BLE frame with different packet fields.
  sc_core::sc_event pscan_evt;
  ……
  void BLE_LL::thread_pscan()
  { ……
    wait( pscan_evt );
    unit32_t winsize;
    ……
    while(1)
    { ……
      struct ble_frame *frame = m_air->rx( winsize );     // where "m_air" is a pointer of "Air".
      ……
      if (frame == NULL)
      break;  ……
      winsize = winsize – win_passed;
      ……
      continue;       ……
    }
  …… }
```

Figure 3.6 is an example of a part of "*pscan()*" mode where the BLE_LL receives a packet during the scan process. The mode begins with opening a scan window (which equals to the Rx chain enabling for the transceiver) with a defined length of "*winsize*" (①). After waiting of a period of time "*win_passed*" it receives a non-null packet (②). We suppose that it is not the end of the scan window, so LL continues to open the scan window with the rest of the window size as "*winsize = winsize – win_passed*" (③④). LL keeps waiting during the rest of the scan window without receiving any other packet until the end of the scan window (⑤), LL then receives nothing in "frame" (frame == NULL) and it may go back to "Standby"

as what the Host tells it. In the case where LL doesn't receive any packet in the scan window, it goes from ① to ⑤ directly.

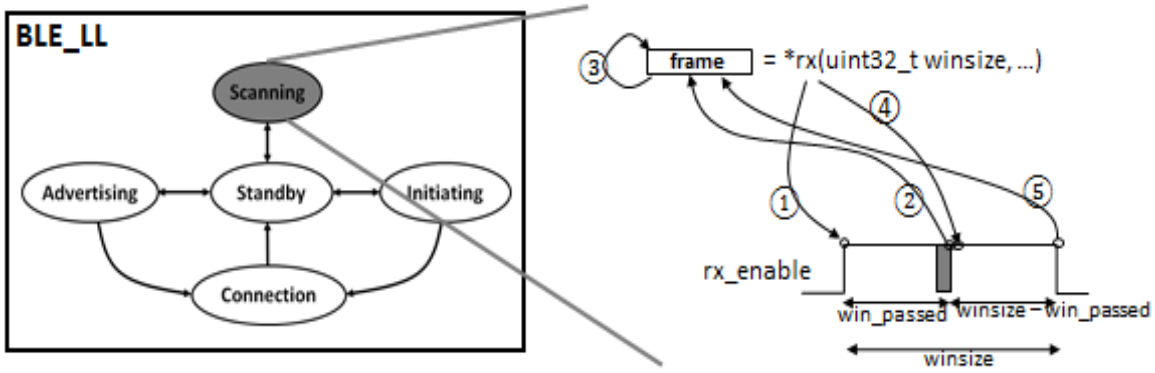


Fig. 3.6: BLE LL passive scan behavior

3.3. Air Modelling

The Air model, presented in Figure 3.7, is the common data channel in this platform. It consists of two main functions "*tx()*" and "*rx()*". They are called in the LL threads each time when there are transmissions or receptions in the simulation, e.g. "*rx()*" function called in the passive scan.

In the high-level simulation, the real data transmissions are not necessary. The packet exchanges are abstracted as sequenced action. Since it is aimed to the high-level error test, the data exchange in this channel is supposed to be perfect which means that there is no PHY level data exchange.

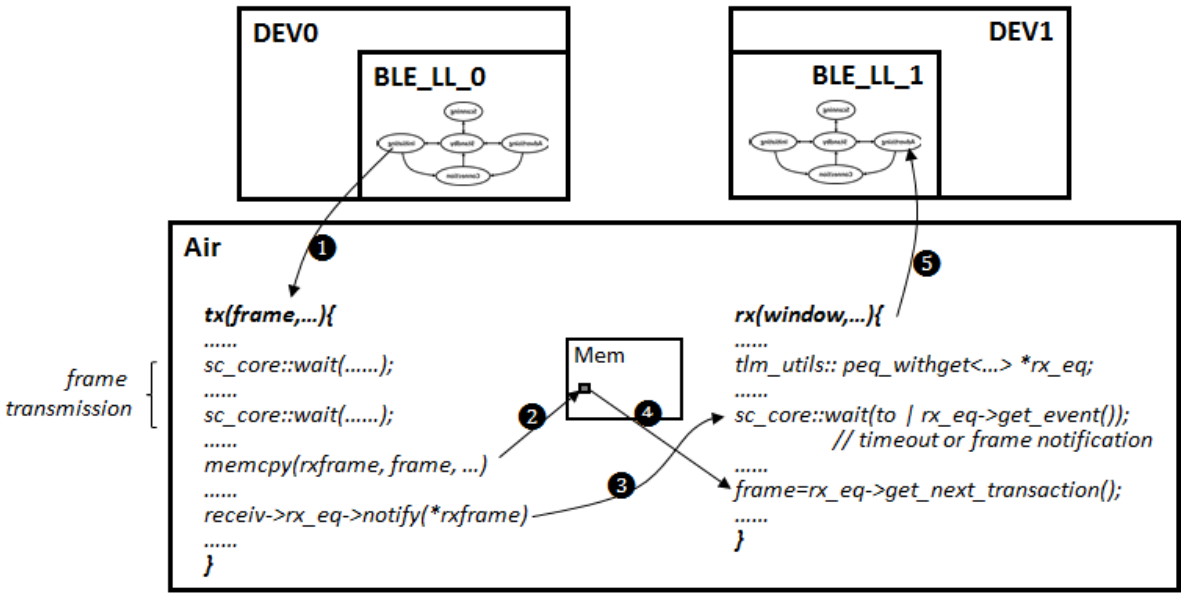


Fig. 3.7: Transaction example with the original "Air"

An example of a transaction between two LLs is used to generally explain the functionality of the channel model in corporation with LL level (cf. Fig. 3.7). DEV0 and DEV1 represent two different BLE devices in this platform. DEV0 sends a packet from a certain LL state and DEV1 receives it in parallel. The "*tx()*" in "Air" simply takes the packet from DEV0 and transmits it to the "*rx()*" called by DEV1. During this process, "*tx()*" is programed to pass the different parts of a BLE frame by "waiting" a defined period of time, e.g. "*sc_core::wait(40,sc_core::SC_US)*" for representing the Preamble and AA transmission. Then the synchronization between the "*tx()*" and the "*rx()*" takes place at the last moment of a frame transmission where "*tx()*" writes this frame into the common memory which is shared with "*rx()*". In the "*rx()*" side, it is just trigged by either an event of timeout or an event of notification to take the frame from the memory. If the timeout has never happened, "*rx()*" will always take a frame from the memory but it could be empty which represents that it doesn't receive anything.

3.4. Interface creation between LL and transceiver

The existing BLE device models, which contain the entire digital part constructed with CPU, Memories, UART, etc, are described in SC-TLM. LL in the digital part is directly communicated with BLE RF part. So, we have now to integrate the transceiver SC-AMS model into the SC-TLM platform.

As shown in Figure 3.8, the transceiver communicates with LL. There should be an interface between them to convert the BLE frame into the timed bits stream or inversely, which actually means the data conversions between the TLM and AMS domains. We call this interface as the "Data path", because beside the interface of the data exchange, there are another kind of signals as the control signals. They come from LL and are used to deliver the control signals (e.g. turn-on/turn-off signals) or configuration information (e.g. RF block configuration as RF specifications (NF, the gain, the power, etc.)) to RF transceiver. As a part of the interface, they are called "Control path".

In the rest of this section, the construction of these two paths will be explained in detail.

3.4.1. Data path

As the transceiver model is written using SC-AMS TDF MoC, an RTL interface should be created in order to realize the data conversion between the two domains. This interface needs to convert the BLE packet from LL into a series of bits of "0" and "1" and put them into the Tx chain with a data rate in the transmission path. And in the reception path, it needs to merge the received data into packet and transmits it up to LL for later processing. As presented in chapter 1 [30], this interface can be easily realized by transforming a BLE packet to a certain data type (bool, int, double, …) and write them to the output every defined time step which is the reciprocal of the data rate.

In this work, we have created the data path which converts the BLE packets to a series of bits and sends them one by one with the data rate of 1 MHz to the Tx input of the transceiver (Tx_data in Figure 3.8). In the other side, we don't have to operate exactly in the inversed way to the Rx chain. In the Rx output, the received data are oversampled bits stream at the end of the demodulator (with the data frequency of 13 MHz). We use a Correlator in the end to detect the received packets. It takes the Rx output oversampled bits stream and stores them in a long register to detect a BLE packet. At the end of the successful reception of one packet, this packet is sent to LL. In the case of a failed reception, LL will receive a NULL packet.

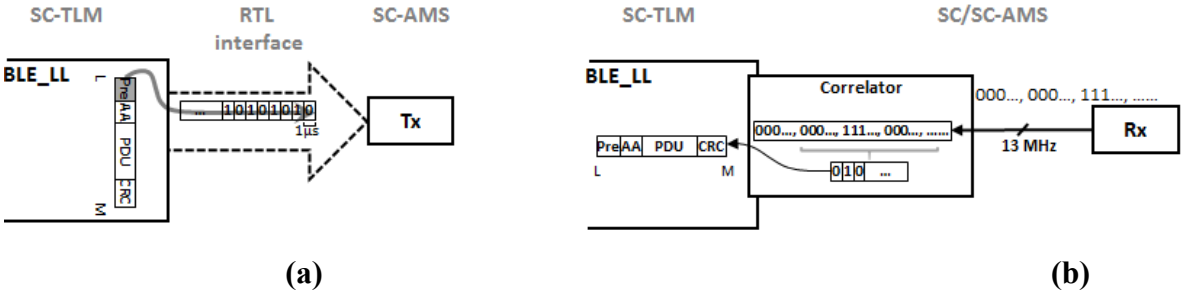


Fig. 3.8: Interfaces between the transceiver and LL for (a) Tx path and (b) Rx path

The bits stream is firstly under-sampled by a factor of 13 (which brings the bit rate back to 1 MHz). Then, the bits are detected by the Correlator to find a certain field of a packet in the order of Preamble, AA, PDU, etc.

It is worth to mention how the Correlator works with the control signal from LL. The long register used to store the Rx output bit stream doesn't shift all the bits with the clock of 13 MHz. This means that in this register, we choose to not move the received bits during the detection process. The long register is used with a finite length which is a little bit longer than 13 times the longest BLE packet length (47 octets).

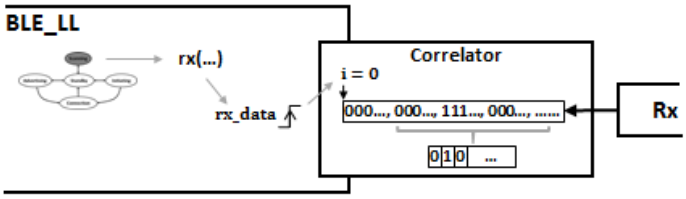


Fig. 3.9: LL controlled Correlator

In the reception process, the Correlator is activated by a control signal coming from LL which is called "rx_data". When LL is ready to send a packet, at the same time it calls the reception function "rx()", the signal "rx_data" is set to "1" and the rising edge of this signal resets the register index "i" in Correlator (cf. Fig. 3.9). Since the register is larger than the longest BLE packet, there won't be overflow. The signal "rx_data" will be set as "0" at the end of the packet, but the register will be reset at the beginning of the next reception.

3.4.2. Control path

Besides evaluating the transceiver performances, we need also to estimate the transceiver energy consumption during BLE applications. So, we have to take into account the power consumption of each block, as a generic parameter of each corresponding model.

The energy estimation permits to understand the global transceiver consumption which is a useful information before the implementation and circuit design. The other objective is to understand the largest energy consumer(s) in the transceiver for a given use case. The RF part is the main energy consumer in a transceiver system. The energy estimation of this part needs to be accurate for each RF block. The block power consumption can be obtained by measuring real circuits, or by extracting this feature from SPICE level modelling simulations. In this work, the power values, which are given by Riviera Waves Company, are coming from measurement of a fabricated transceiver circuit.

Moreover, to optimize the energy consumption, a power gating technique is applied and implemented in the transceiver. Then, RF blocks are switched on only when it is necessary in order to avoid the waste of energy consumption during the communication. This command is realized by a RF control component called "Sequencer" in the transceiver system which is treated as a part of the interface between the transceiver and LL.

Band of registers for delays

	L								M
txup_dly[8]	5.02	34.96	0.17	93.98	0.17	10.04	0.17	0.17	0.17
txdn_dly[8]	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17
rxup_dly[8]	0.17	44.83	4.33	4.33	6.06	0.17	0.17	0.17	0.17
rxdn_dly[8]	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17	0.17

Band of registers for outputs

	L								M
txup_out[8]	0	1	3	7	15	31	31	31	31
txdn_out[8]	15	7	3	1	0	0	0	0	0
rxup_out[8]	0	1	3	7	15	31	31	31	31
rxdn_out[8]	15	7	3	1	0	0	0	0	0

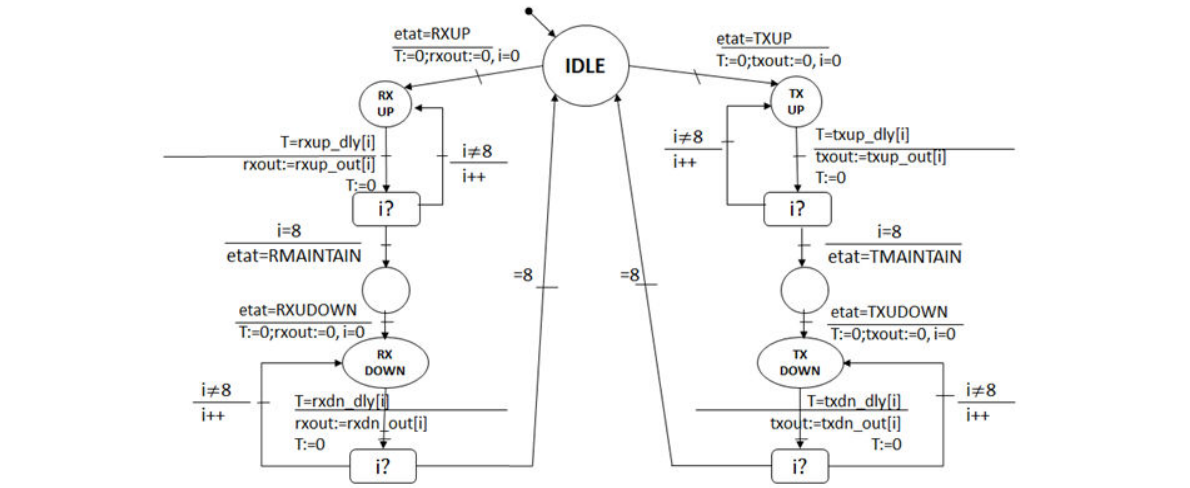


Fig. 3.10: Sequencer FSM with the RF control registers

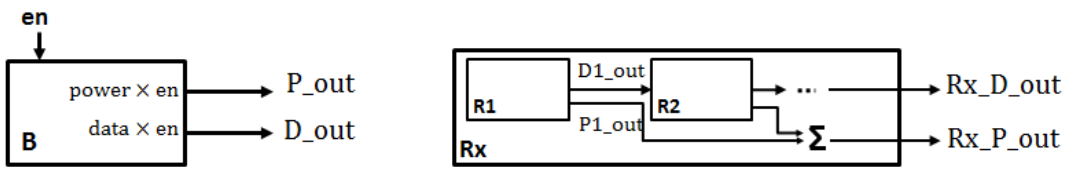
The main content of the Sequencer is shown in Figure 3.10. It contains mainly two banks of registers to store the delay values and the output values. These values are pre-defined and are reprogrammable. The delays and outputs are separated in 4 sub-groups respectively for the 4 different processes of the beginning and the ending of the Tx activation and the Rx activation separately. They are placed in order with 8 steps in the registers (cf. Fig. 3.10). Each delay value corresponds to an output value which has the same register index number as this delay value. Their usage will be explained with the operation of the Sequencer FSM.

The Sequencer is modelled as a "*sc_module*". In the Sequencer, we define a variable, which is driven by LL, to represent the different states called "*etat*". The 7 possible values of this variable and their representations are given in Table 3.2. When there is an emission or a reception, "*etat*" will changes from "*IDLE*" to "*TXUP*" or "*RXUP*" respectively. It makes the state machine advance to the Tx or Rx operation.

Table 3.2: Sequencer states and their representations

"etat" values	*IDLE*	*TXUP*	*TXDOWN*	*RXUP*	*RXDOWN*	*TMAINTAIN*	*RMAINTAIN*
"etat" values representations	Idle	Tx activation start	Tx activation end	Rx activation start	Rx activation end	Keep the final Tx activation	Keep the final Rx activation

We will illustrate this operation for a Tx operation. When LL needs to activate Tx chain, it makes "*etat*" equals to "*TXUP*". Then, the Sequencer enters to the "*TXUP*" state where it begins to read the delay values in the order given in the registers. It initials firstly a counter "i" to represent the registers index, and it will go through the register "*txup_dly[8]*" to read the delay values with a loop of 8. It begins with the cell of "*txup_dly[0]*" which corresponds to the value of 5.02 which means 5.02 times of a defined time step "*Tstep*", i.e. $T = 5.02 \times Tstep$. The state machine will wait for "T" seconds then outputs the cell "*txup_out[0]*" value. Then, the 4 registers for outputs, presented in Figure 3.10, control the power gating. For example, the first bit of the 5th byte of the register txup_out[8] is equal to 1, corresponding to switch on the PLL. Since the RF blocks contains their own accurate power consumptions, the control signals are used to enable the output data and the output block power as shown in Figure 3.11 (a). Then as an example in Figure 3.11 (b), the Rx chain gives instantaneously the precise power consumption at Rx output by adding all the instantaneous block powers.



(a) **(b)**

Fig. 3.11: (a) RF single block and (b) Rx chain with outputs controlled by the Sequencer

When the state machine in Figure 3.10 iterates the "*txup_dly[8]*" registers, the state machine has activated the entire Tx chain in the transceiver. It sets "*etat*" to "*TMAINTAIN*" to maintain the last output control signals for Tx, until LL makes "*etat*" to "*TXDOWN*", indicating the end of an emission. The following Tx deactivation operation is with the same principle as the Tx activation.

Figure 3.12 gives an explanation of the Sequencer functionality with a BLE packet emission and Figure 3.13 presents this simulation with power output curves. In the figure, Tx1 is sends a BLE packet. "Tx1_Power" corresponds to the instantaneous Tx power output of this transaction. The output power begins to rise after "*etat*" is set to "*TXUP*", and it varies step by step according to the plan of the Tx chain activation defined by the registers in Figure 3.10, and the same for the Tx chain deactivation when "*etat*" is set to "*TXDOWN*". The energy estimation can be obtained by integrating this curve at the end of a global simulation which will be shown later.

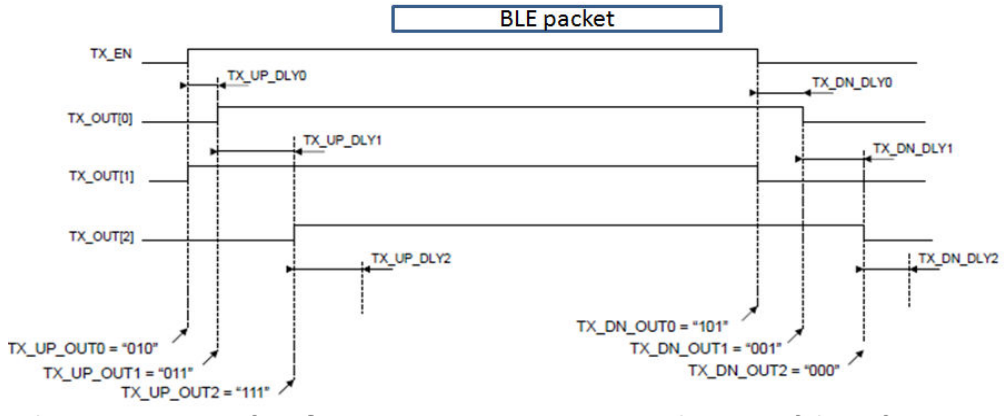


Fig. 3.12: Sequencer functionality

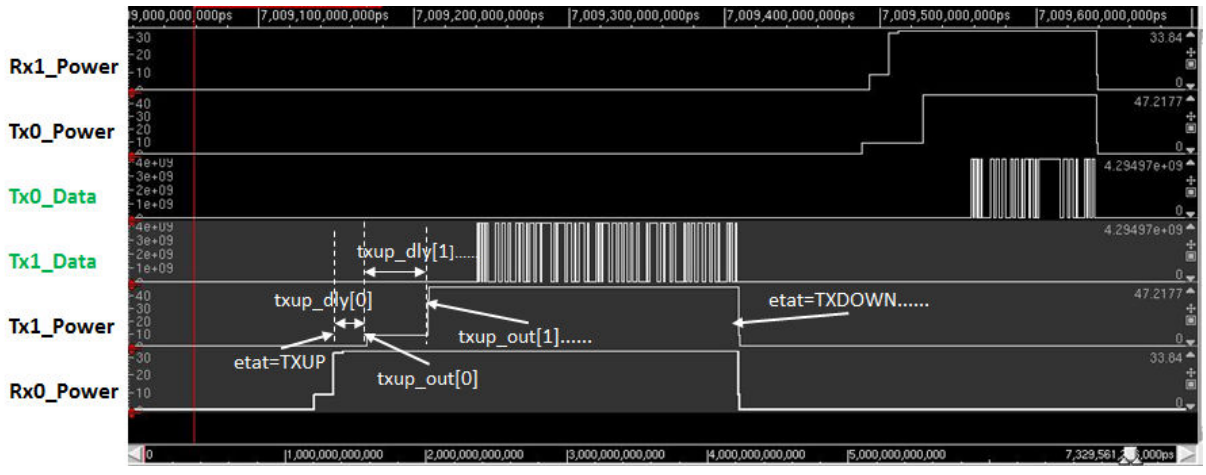


Fig. 3.13: Transceiver power output sensitive to the control signals

The energy estimation can be obtained by integrating this curve at the end of a global simulation. Since the energy result is more obvious from longer use cases, it will be shown in the future sub-sections.

3.5. Transceiver configuration and integration in TLM system

After the previous preparations, the transceiver model can be finally initiated with the example of configurations given in Table 3.3, and then integrated into the SC-TLM simulation platform. The Tx chain model is maintained as the original version since it is not the objective of this work. In order to estimate the Rx chain performance, the Rx chain model is constructed with the configurable models, built in the previous sub-section with the interface which contains the data path ("*Tx_data*" and the correlator in Figure 3.14) and the control path (Sequencer in Fig. 3.14).

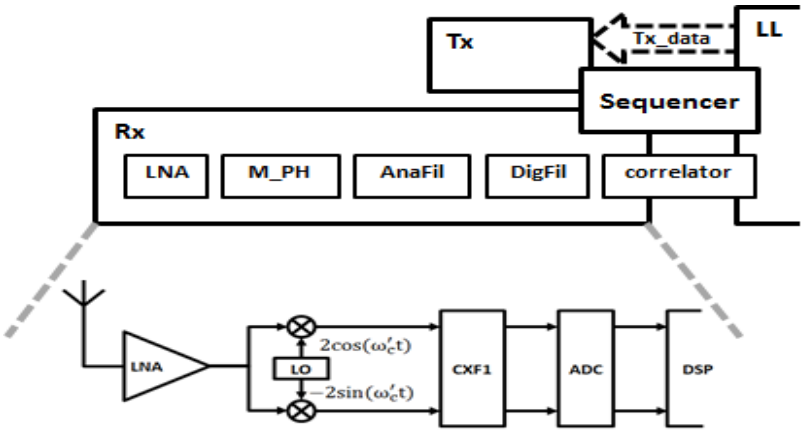


Fig. 3.14: Transceiver integration into the TLM system

Table 3.3: RF Front-end performance optimization configuration example

	LNA	Mixer	CXF	ADC
Gain(dB)	20	6	20	
NF(dB)	5	13	15	39.2
IIP3 (dBm)	−15	25	−41	
Phase Noise	−125 dBc/Hz (≥ 3 MHz)			

An example of the powers of the different parts of the transceiver is given in Table 3.4. These values of consumptions correspond approximately to a circuit design by Riviera Waves Company. For reasons of confidentiality, we modified a little bit these values, while keeping the order of magnitude. These values will be used for the following simulations.

Table 3.4: Transceiver power consumption configurations

Tx	Txbias	Txpll	TxLO	Tx_modem	TxfiltI	TxfiltQ	TxdacI	TxdacQ	Txmix	Txpa
(mA)	0.34	9.64	3.58	2.13	2.0	1.93	0.3	0.26	19.0	11.2
Rx	Rxlna	Rxmix	Rxfilt	Rx_modem	Rxadc	RxLO	Rxpll	Rxagc	Rxbias	rssi
(mA)	2.3	0.82	1.93	10.2	5.1	5.13	9.64	1.37	0.11	0.6

3.6. Global simulation of BLE system

3.6.1. Introduction

The global simulation of this work is for the transceiver energy consumption and the Rx performance estimation. The platform is constructed as the simplest architecture as a network with only 2 BLE devices.

3.6.2. Simulation platform construction

Since we focus on the RF transceiver performance, we suppose that there is no high level error other than the errors due to the transceiver.

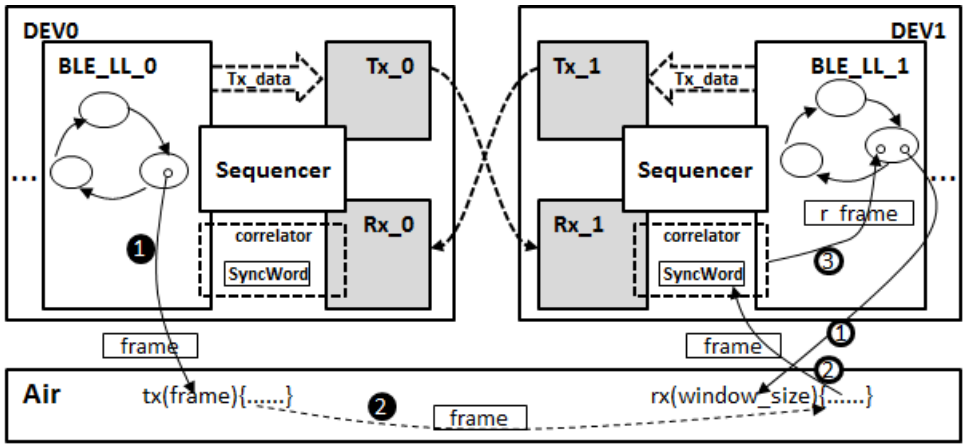


Fig. 3.15: BLE network of 2 BLE devices

As shown in Figure 3.15, after the transceiver integration, the network is created with two devices called "*DEV0*" and "*DEV1*". Since we don't analyze the channel performance, we suppose that these two devices will only communicate with each other and we connect directly "*Tx_0*" with "*Rx_1*" and "*Tx_1*" with "*Rx_0*".

The original "*Air*" is still reserved as a virtual channel in this simulation as it can simplify the reception process. The "*Air*" block is synchronized by the transceiver timing. This block put the ready-to-send packet to the target Rx. So, we have to use it in order to transmit the "SyncWord" in the "Correlators" directly without using the LL communication where it will create higher level errors. During a reception, the received AA (by going through the SC-AMS transceiver path) will be verified with this "SyncWord".

When "*DEV0*" sends a packet through the data path, which is in the itinerary of "*BLE_LL_0*", "*Tx_data*", "*Tx_0*", "*Rx_1*", and "*BLE_LL_1*", it sends directly this packet through "*Air*" (❶). "*Air*" passes directly to the Correlator of "*DEV1*" with respecting the predefined timing(❷). This operation is synchronized with the SC-AMS modelled data path (the transceiver). In the other side, "*BLE_LL_1*" of "*DEV1*" takes the "AA" as "SyncWord" in its "Correlator" at the beginning of its Rx process. This process is actually in parallel as the

Tx process of DEV0 (①②). The real packet transaction goes through "*Tx_data*", "*Tx_0*", "*Rx_1*" and finally it enters to the "Correlator" of "*DEV1*". Once the "AA" received from this data path matches the "SyncWord", the Correlator detects the whole packet and then sends it to "*BLE_LL_1*" (③). It stores also other fields of the packet in Correlator for BER estimation by comparing them with the received data through the data path.

3.6.3. BLE use cases and Energy estimation

To realize a simulation in order to estimate the energy consumption, we need firstly a use case which can be a BLE application or a customized scenario with respecting the BLE communication standard. As the BLE applications are normally very short in time, we choose to define a scenario where there is a lot of data exchange.

Use case 1

The first use case applied for the global simulation is very short. We use this example to verify the functionality of the simulation for the energy estimation. The scenario content is shown in Figure 3.16 and is explained below.

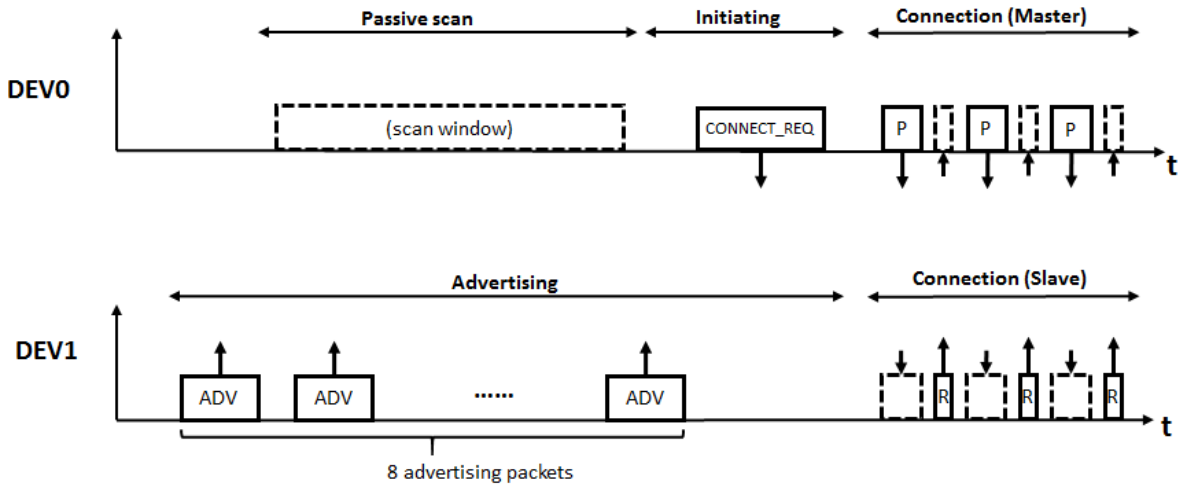


Fig. 3.16: First use case scenario

1. DEV1 enters firstly in Advertising mode and sends ADV packets in advertising channel 37 regularly (an advertiser should send ADV packets regularly in all the advertising channels as 37, 38, 39 according to the BLE standard, but here the scenario is simplified for DEV1 sends ADV packets only in channel 37). DEV0 enters firstly in Scan mode and carries out a passive scan. It opens a scan window to listen on channel 37 and it receives 6 ADV packets sent by DEV1 during the scan window.
2. At the end of the scan window, DEV0 comes out from the Scan mode and enters to the Initiating mode to ask DEV1 for an establishment of connection by sending it a Connection requirement (CONNECT_REQ), and then it enters in Connection mode

with the role of Master. In the other side, after the 8th ADV packet sending, DEV1 receives the requirement. It comes out from the advertising mode and enters the Connection with the role of Slave. The connection is established at this moment.

3. In the connection mode, DEV0 realizes three emission of data packets ("P" packets in Fig. 3.16) and DEV1 receives them in success each time and sends a response ("R" packets in Fig. 3.16) to DEV1.

3.6.4. Simulation and energy consumption results

Then, we run the simulation with Rx noise floor of -114 dBm. The power level of the input signal will vary from -96 dBm to -91 dBm, in this simulation. The two objectives of this simulation is to verify the transceiver energy consumption functionality, and to obtain the transceiver BER performance hence the RF receiver sensitivity. Figure 3.17 presents the simulation result of the energy estimation for use case 1.

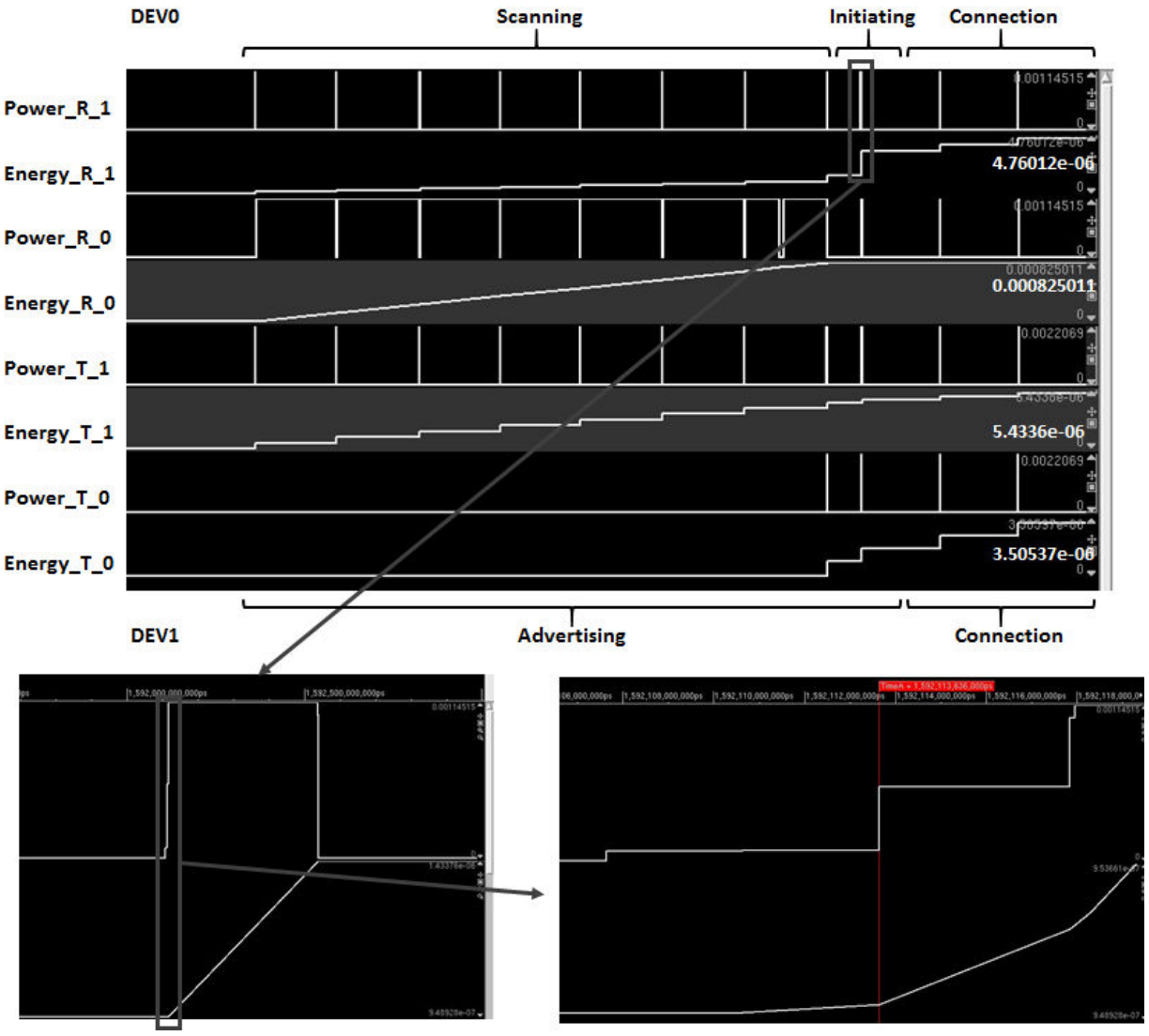


Fig. 3.17: Energy estimation of use case 1

In Figure 3.17, the curves marked as “Power_” represent the instantaneous power of the transceiver (Tx or Rx) of “DEV0” or “DEV1”. Since the power curve follows exactly the Tx or Rx enabling signal, it can be used to generally observe the data exchange between the two devices. So, we can notice that the simulation result corresponds exactly to the scenario of use case 1. We can see from the details shown in this figure that the energy curve is the power integration over time with respecting the transceiver activation process, so it is accurate to the transceiver block level.

The performances of the transceiver in terms of BER are exactly the same as presented in chapter 2 (cf. Figure 2.42). This is due to that we have not changed the generic parameters of the different blocks of the transceiver.

Other use cases

The first use case is too short to observe in detail the energy consumption. It will be not evident to obtain the consumption difference between the different transceivers. Other longer use cases are applied. In the second use case, two devices take a short period to establish a connection. Then after they are connected, “DEV1” sends 100 data packets to “DEV0”.

The simulation output curves are shown in Figure 3.18 and the energy results are listed in Table 3.5. This result gives, for the first time, the energy estimation for a use case long enough. It will be used to compare with the optimized transceiver energy consumption in the following sub-section.

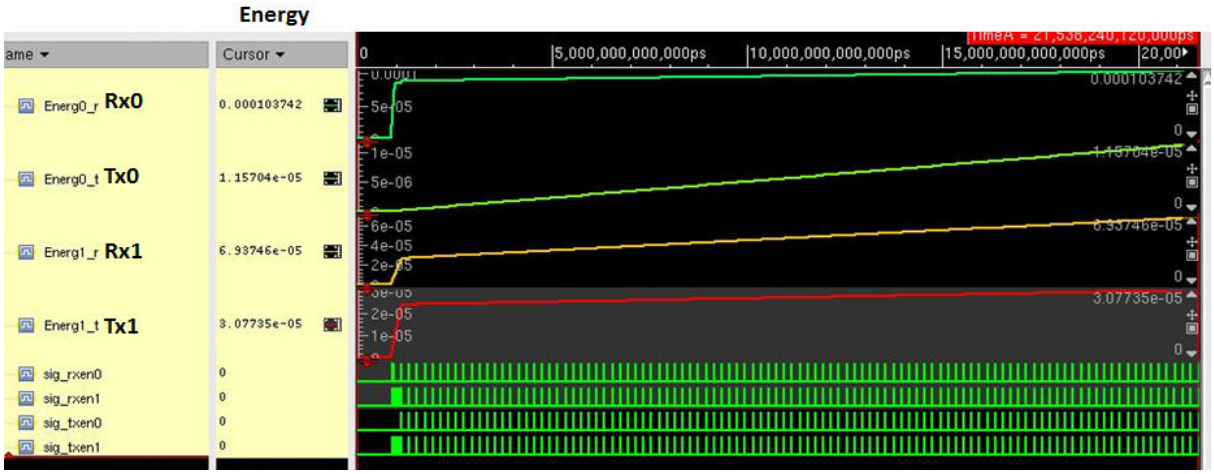


Fig. 3.18: Energy estimation of use case 2

Table 3.5: Energy estimation results of use case 2

Tx_0 energy	1.16e-5 J
Rx_0 energy	1.04e-3 J
Tx_1 energy	3.08e-5 J
Rx_1 energy	6.94e-5 J
Simulation time	54 mn

Use case 3 defines the two devices carry on several Advertising channel events. The simulation output curves are given in Figure 3.19 and the energy results are listed in Table 3.6.

Table 3.6: Energy estimation results of use case 3

Tx_0 energy	4.51e-6 J
Rx_0 energy	9.14e-6 J
Tx_1 energy	1.41e-6 J
Rx_1 energy	4.78e-3 J
Simulation time	22 mn

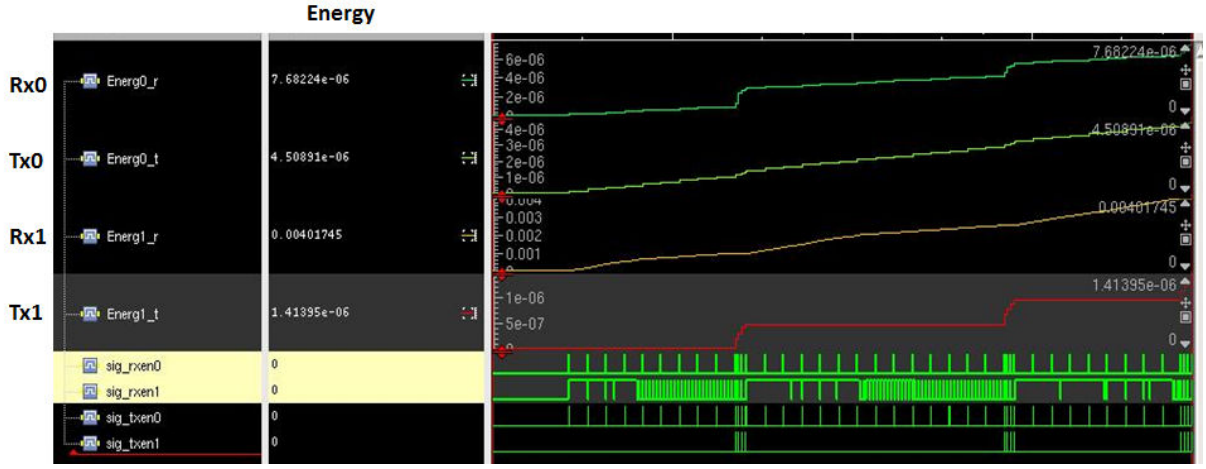


Fig. 3.19: Energy estimation of use case 3

Like the previous use case result, this result will be used as a second use case to compare with the optimized transceiver energy consumption in the future sub-section. We have finished so far the work of modelling and verification of the BLE simulation platform in SC-TLM and SC-AMS targeting to the transceiver performance and energy consumption estimation in the circuit level. The simulation is extremely faster compared with the traditional methods. The next step is to compare the performance of these two transceivers by reapplying the simulation to an optimized transceiver.

3.7. Example of application: optimization of the transceiver energy consumption

3.7.1. RF system optimization

Since the RF circuit is the most energy consumption part in a communication system, we always want to find a transceiver architecture which consumes less. The RF designer tries to adjust from the RF parameters to achieve certain purpose, e.g. sacrificing the power consumption or the system complexity for better system performance.

In this work, we want to optimize the energy consumption of the original transceiver while keeping the original transceiver system performance in a certain threshold. In this case the optimization flow can be described in Figure 3.20.

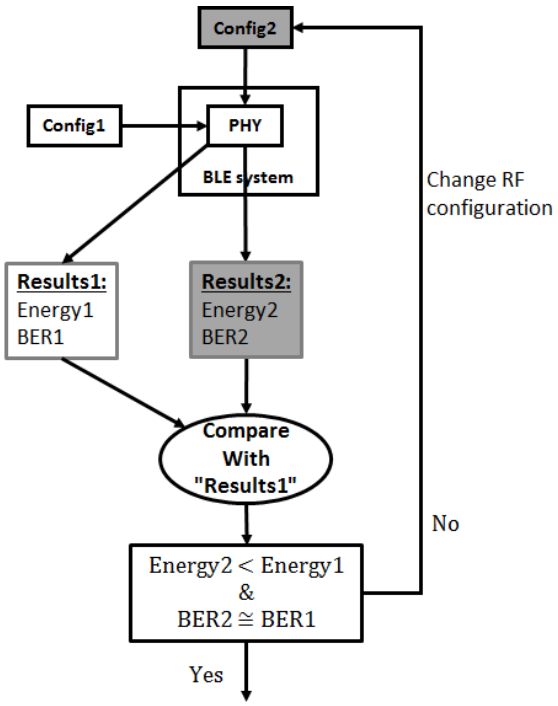


Fig. 3.20: RF system optimization with the global simulation platform

In the design flow, we need firstly the performance of the original transceiver system ("Config1" in Fig. 3.20) as a reference. In this work we need to reduce the system consumption and maintain the system BER as defined in the BLE standard, so we choose these two parameter as the simulation results. Then we modify one or several RF blocks or the whole transceiver architecture to try to reduce the global energy consumption. Based on this modification, a second system configuration ("Config2" in Fig. 3.20) is obtained and will be used as the input of the global BLE system simulation. It will generate a second result contains the new transceiver performance as "Energy2" and "BER2". This result will then be compared with "Results1". If the consumption is reduced to a level which matches our need and the BER respects the communication standard, the new configuration can be used as the start of a new design based on the original one. Otherwise we still need to modify the configuration and refollow the design flow in the figure.

This design flow requires the transceiver models easy to be modified and a simulation running fast, which corresponds to the previous platform we have constructed.

3.7.2. RF chain with new configuration example

As an optimization example, we choose to change Rx blocks in this work. To maintain the Rx performance, we can choose certain blocks which have different gains and noise

figures of their own but contribute generally the same gain and noise figure in total to the Rx chain, but in parallel they consume generally less power.

In the Rx chain, we decide to change the 2 blocks of LNA and Mixer. To simplify the comparison with the original Rx configuration, they are listed together with the new LNA and Mixer where the new gains, NFs and power are written in boldface in Table 3.7.

Table 3.7: Rx Front-end performance optimization with different LNA and Mixer

	LNA		Mixer		CXF	ADC
Gain(dB)	20	**40**	6	**0**	20	
NF(dB)	5	**3**	13	**17**	15	39.1
IIP3 (dBm)	−15		25		−41	
New power (mA)	**2.01**		**0.01**			
Phase Noise	−125 dBc/Hz (≥ 3 MHz)					

.

New results are obtained by rerunning the simulation. The simulation is run separately for the BER estimation and the energy estimation for the reason of saving time since the energy estimation needs only one time, but the BER estimation needs simulations with the varying input signal power.

3.7.3. Simulated results

The simulation of the BER estimation is run with sending 1000 packet of 40 bits from a Tx to an Rx. Rx noise floor is set as −114 dBm and input signal power level as a variable from −96 dBm to −80 dBm. The BER curve is shown in Figure 3.21. The new sensitivity is listed in Table 3.8, with the original one.

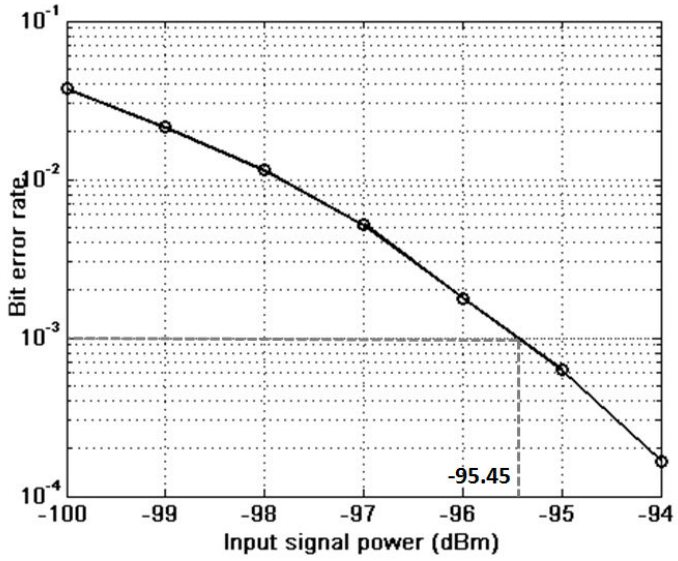


Fig. 3.21: BER and Rx sensitivity simulation results

Table 3.8: BER result of the new transceiver

$\mathbf{P_{in}}: -100\ \mathbf{dBm} \sim -94\ \mathbf{dBm}$ and $\mathbf{Noise\ Floor} = -114\ \mathbf{dBm}$		
Sensitivity	Old transceiver	-93.25 dBm
	New transceiver	-95.45 dBm

This example shows that this new optimized design is correct and efficient if the requirement in terms of Rx sensitivity was -95 dBm, The simulation platform is suitable to validate this kind of exploration.

The simulations for the energy estimation are run with the same use cases as used in the previous section. We use the last two scenarios, and the simulation results are listed in Table 3.9, with the results of the original transceiver.

Table 3.9: Energy consumption of the new transceiver

Use case 2	Old transceiver	Tx_0	1.16e-5 J	Use case 3	Old transceiver	Tx_0	4.51e-6 J
		Rx_0	1.04e-3 J			Rx_0	9.14e-6 J
		Tx_1	3.08e-5 J			Tx_1	1.41e-6 J
		Rx_1	6.94e-5 J			Rx_1	4.78e-3 J
	New transceiver	Tx_0	1.16e-5 J		New transceiver	Tx_0	4.51e-6 J
		Rx_0	**8.72e-5 J**			Rx_0	**7.68e-6 J**
		Tx_1	3.08e-5 J			Tx_1	1.41e-6 J
		Rx_1	**5.83e-5 J**			Rx_1	**4.02e-3 J**

The results show that the optimized RF transceiver consumes 16% energy less than the old one and the Rx sensitivity is also optimized of 2.2 dB.

3.8. Conclusion

In this chapter, the configurable models of the RF receiver chain are developed. Then the transceiver based on these models is built and integrated into a SC-TLM described BLE system. To be able to connect the SCAMS domain with the SC-TLM domain, the interfaces as the data path and the control path are created. Then, we have realized a global simulation for a 2-device BLE network with different use cases, which determines the energy consumptions of the Tx and Rx chain of each device. Finally, an optimization work based on this result is realized by easily replacing the Rx LNA and Mixer with the new ones (design by the Riviera Waves Company) containing optimized parameters as NF and Gain. The simulation is redone and gives the after-optimization results as the new transceiver BER and new energy consumptions which are both better than the old transceiver.

4. WSN modelling and simulation

4.1. Introduction

We have up to now simulated a point-to-point communication between two BLE devices. The objective in this chapter is to show that our work can be extended to a more complete system, e.g. a sensor network.

Figure 3.22 shows an example of wireless scenario with three nodes interacting together through the same radio channel [30]. Each node consists of several components connected to a bus, i.e., CPU executing software (application, protocol stack, drivers), memory, ASIC for baseband communication functions and the RF circuit. Also the communication channel must be considered during simulation since it affects data transmission and thus system performance.

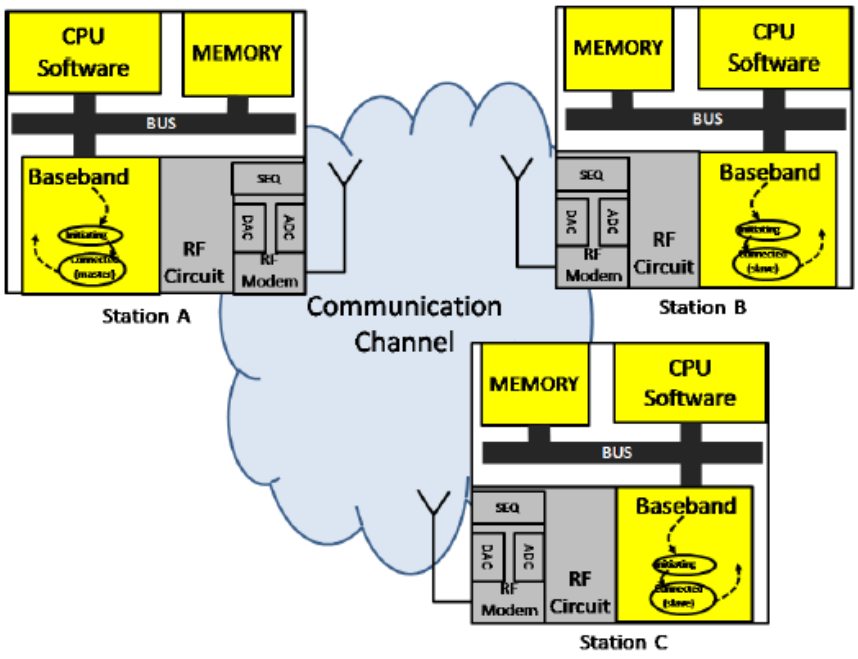


Fig. 3.22: Wireless sensor network scenario

The PHY and Data link levels modelling are realized with the work described in the previous chapters. The network and upper levels modelling will be presented in the rest of this chapter. Figure 3.23 shows a suitable model of communications at Datalink Level. Each station model should be connected to the model of the medium through an output signal to send bits, an input signal to receive bits and an input signal to test if the channel is busy. The last one is an abstraction of the various techniques used to assess the state of the channel. The model of the medium should reproduce the behavior of a shared binary channel by using a Resolver function which takes all the transmitted input signals and decides the logic value to be read by the stations (Data out) and the state of the channel (Carrier out).

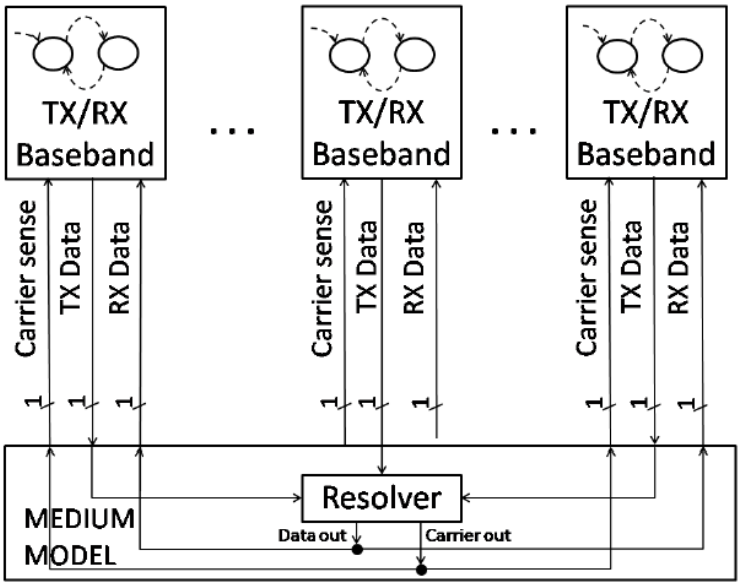


Fig. 3.23: Datalink level modelling

Event-driven model of computation is the most suitable approach to model communications at Datalink Level since the scenario is quite similar to a shared bus in digital systems. Hardware description languages like VHDL, Verilog or SystemC-RTL can be used as in [31]. The event frequency is proportional to the bitrate.

4.2. Network-level modelling

The main objective of Network Level is Routing, i.e., the transmission of a PDU over a path composed of set of intermediate nodes and links. Complex topologies with multiple links are addressed in the routing process. Nodes can be classified into "hosts" which contain user applications and "routers" which route PDUs over the network.

At this level, the details to be modelled are delays and packet loss due to congestions (i.e. a packet is discarded when it enters a full queue). Figure 3.24 (a) shows a simple but complete model for this level in which hosts (denoted by H_{ij}) exchange PDUs by using routers (denoted by R_i). Connections are modelled by unidirectional links with queues. Each link is characterized by a capacity and a delay. Each queue is characterized by the size in bytes. Hosts and routers have one queue for each output interface. The delay between two adjacent nodes is the sum of the time to traverse the output queue, the time to encode the packet (which depends on packet size and link capacity), and the link delay.

The most efficient way to simulate Network Level scenarios consists in generating events for each packet (i.e. entering/exiting a queue or a link). The event-driven model of computation should be used as done in well-known packet-based network simulator like NS-2 [32], OMNET [33], OPNET [34], and SCNSL [35]. The event frequency is proportional to the packet rate.

The main objectives of the levels above Network are flow control, packet ordering, reliability, common data representation, end-to-end security, and application logic. At this level, the details to be modelled are the content of exchanged messages and the transmission delay. As depicted in Figure 3.24 (b), these functions can be represented as event-driven processes exchanging messages over a bi-directional full-duplex point-to-point link which is characterized by the capacity and delay of the end-to-end connection between the two processes. The most suitable model of computation for this scenario is event-driven at transaction level; in particular, SystemC-TLM can be effectively used for this purpose as described in [36].

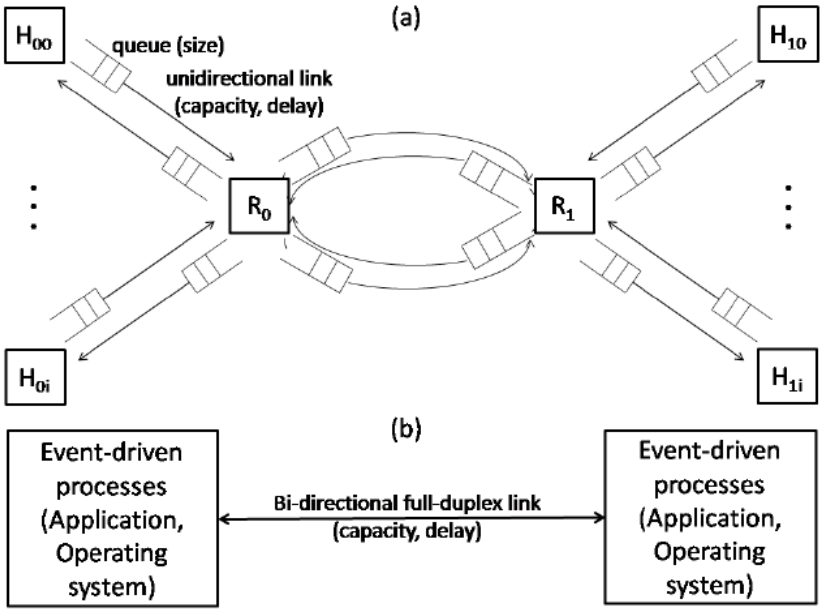


Fig. 3.24: Modelling of the network level (a) and of the upper levels (b)

4.3. Model abstraction

The techniques presented in chapter 1 (cf. Fig. 1.32 in Section 4.6) can be used to mix the abstraction levels. Even if the same SystemC simulator is used to implement all the involved models of computation, simulation speed is affected by this approach since it is limited by the slowest model of computation; furthermore, there is a computational overhead for synchronization between the different models of computation.

Another promising approach consists in using abstraction. Simulation models at a higher abstraction level are enriched with the high-level effect of details belonging to lower abstraction levels without simulating them in the same runtime session. For instance:

1) at Physical Level, the effect of non-linear component behavior at carrier level is reproduced in baseband equivalent simulation;

2) bit error statistics found at Physical Level is used to corrupt bits at Datalink Level, Network Level, and Upper Level;

3) delay statistics found at Datalink Level (because of the medium access control) is used to delay packets at Network Level and Upper Level;

4) packet loss statistics found at Network Level (because of congestions) is used to drop messages at Upper Level.

Techniques (2) and (3) can be implemented by using the SystemC Network Simulation Library (SCNSL) [35] whose architecture is shown in Figure 3.25. Tasks are used to model node functionality, i.e., packet production and consumption. Tasks are hosted by Nodes, which are the abstraction of physical devices. Thus, tasks deployed on different nodes shall communicate by using the channel model. The Channel represents the transmission medium, and SCNSL provides models for both wired (full-duplex, half-duplex and unidirectional) and wireless channels. In this work the wireless channel has been used to reproduce packet collisions and path loss. The Communicator is a SCNSL component implementing a packet-processing interface. New capabilities and behaviors can be easily added by extending this class. Communicators can be interconnected with each other to create packet-processing chains. Hence transmitted packets will move from the source Task to the Node, traversing zero or more intermediate communicators, and then they are transmitted by using a channel model, and cross the communicators between the destination node and the destination Task. This way, a modification of the transmission behavior is possible by creating a new communicator and placing it between tasks and nodes. For instance, in Figure 3.25, there are two communicator instances implementing a medium access control (MAC) protocol and other two communicator instances to corrupt packets. Packets from Station 0 to Station 1 traverse the simulator blocks following the red continuous arrows.

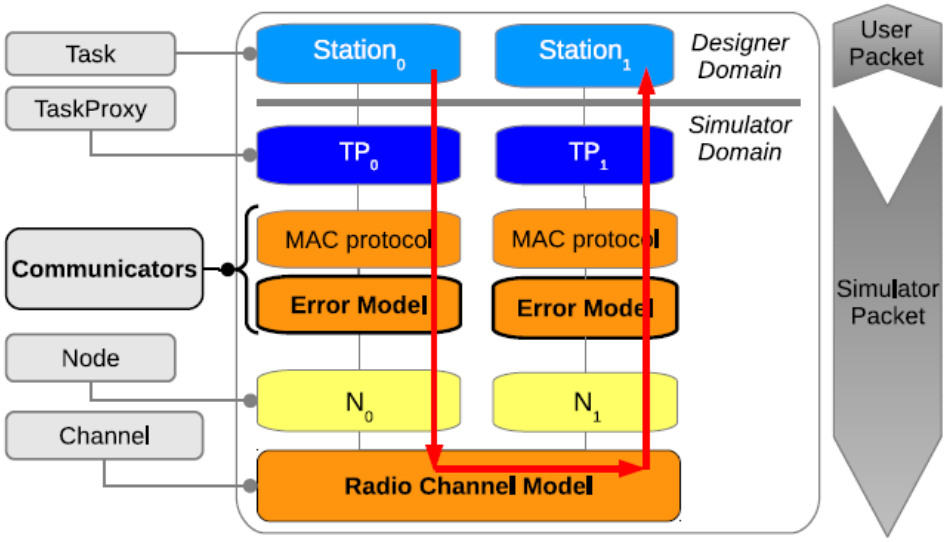


Fig. 3.25 : SCNL internal architecture

Communications are simulated by considering events related to packet transmission and reception. Therefore, the simulation can consider at Network Level. However, communicators are used to implement a bit error injection mechanism according to the curves obtained by the

Physical Level simulation. Furthermore, task interface also provides a signal to check carrier on the network. This feature is typical of Datalink abstraction Level.

4.4. Simulation results

To show the use of different abstraction levels and their mixing to support the design of a wireless device, the case of a BLE network is considered. The functional model provides a high level description that can be used for design space exploration including protocol analysis, performance analysis and architecture exploration. Experimental results are reported about BER and the simulation time of the various modelling approaches.

At Application Level, the transmission of packets is modelled by using SystemC-TLM. Each transaction handles the transmission of a packet. The first line of Table 3.10 reports the simulation speed and the BER at this application level. At Network Level, the model of a shared channel is used. The simulator reproduces the start and end of transmission of each packet at both ends of the channel. The second line of Table 3.10 reports the simulation speed and the BER. Clearly, at these levels (first and second lines of Table 3.10), we cannot take into account the effect of channel impairments and therefore BER is not available.

As reported in the point-to-point system simulation, the simulation time is critical. A new Network Level simulation has been performed by using SCNSL [35] and Communicator instances injecting bit errors in the packet payload according to the statistics on communication performance obtained at Physical Level. Performances are reported in the last lines of Table 3.10.

Table 3.10: Simulation speed at different abstraction levels

Abstraction level	Simulation time	BER
Application Level	0.002 ms/bit	n.a.
Network Level	0.007 ms/bit	n.a.
Network Level + Error model	0.021 ms/bit	0.00315
Network Level + MAC protocol	0.022 ms/bit	n.a.
Network Level + MAC protocol + Error model	0.028 ms/bit	0.00315

The simulation time is slightly higher than pure Network Level simulation but much lower than Physical Level simulation; BER is quite accurately reproduced. This increased speed and the presence of a pure event-driven model give the opportunity to explore the introduction of protocols, e.g. for medium access control (MAC).

Last two lines of Table 3.10 show the increased overhead without and with bit error injection, respectively.

5. Conclusion

In this chapter, we have defined the approaches for the abstraction higher than data link levels for more complete wireless network system modelling. Simulation results show that the use of such approaches in different phases of the design flow allows to test applications and protocols in realistic scenarios is flexible and time saving to adapt different simulation requirements. So that problems and possible optimizations can be evaluated before the actual deployment, thus shortening the time-to-market of wireless systems.

Different realistic use cases have been simulated, and finally we have compared the performances of two transceiver designs realized by Riviera Waves Company. The simulated results, validated by measurements, have shown both BER and power consumption of the new designed are better. We have concluded this chapter by the integration of our platform in a higher level of abstraction (i.e. modelling and simulation). We have interfaced our SystemC and SystemC-AMS model in SCNSL platform for network simulation.

Conclusion and Trends

1. General conclusion

We have presented in this thesis a method of fast simulating complete wireless systems of mixed-signals based on the MIM development method. And this kind of simulation based on system-level modelling can be used for the different abstraction level analysis and system optimization.

This work focus on a BLE RF transceiver modelling and system performance analysis based on a global SC-TLM described BLE (or BT) platform which contains the whole higher-than-PHY levels.

We began with the study of the methodologies concerning the modelling approaches and the RF system considerations. In chapter 1, among the common used modelling approaches for mixed systems, we have chosen the MIM combined with the baseband equivalent modelling method to realize an RF transceiver modelling in system-level in order to achieve a low-level accurate simulation with a system-level speed. This method requires to choose only the main RF block specifications from the low-level design. So, the RF specifications like linear gain, non-linearity and noise existing in RF system are introduced. BLE technique, especially the link layer level, is introduced as a preparation of the further connection with our transceiver model to achieve the final global simulation. In the end of this chapter, we have introduced the common used modelling tools and languages for mixed signal modelling and finally we have chosen SC-AMS for our transceiver modelling tool, due to the efficiency and the integrity of SC toolset and the adaptation to the existing BT high-level models.

In chapter 2, a complete BLE transceiver SC-AMS model has been functionally realized. Then, the Rx RF Front-end has been refined with the main RF specifications coming from real circuit measurements offered by Riviera Waves company. A simulation speed optimization of 80% has been done after the functional model simulation. The refined model simulation gave the result as the relationship between the BER and the Rx input signal power (i.e. SNR).

In chapter 3, we have firstly developed the configurable models for the Rx RF blocks. Then the SC-TLM model of the BLE LL and the original Air model are presented in details for preparing the LL/PHY interface creation. The interface, which was separated into the data path and the control path, has then been created, The BLE platform has been completed by connecting the SC-TLM domain with our SC-AMS transceiver model, where the transceiver Rx RF was built with the configurable models. The simulation of this platform has been launched with BLE use cases and gave some results as the transceiver RF energy consumptions. Finally, in order to optimize the Rx performance, LNA and mixer blocks have been replaced by new designs with better power consumption. The simulation has been rerun and the Rx energy consumption reduced 16% in comparing with the old Rx, and the sensitivity was 2.2 dB better than the old Rx.

To conclude this chapter, we have presented advanced utility of our modelling method for higher wireless systems modelling. For the different abstraction levels, the simulated results give close BER values with suitable simulation time.

2. Perspectives

In the RF modelling work, the PLL system was abstracted as a phase noise which was only described by an assumed model. To obtain more accuracy, PLL can be better modelled as a complete system with all sub-blocks in detail.

In this work, the interfaces were made only for this BT system. To make the interface more common, we can modify the LL side for a simpler connection with the transceiver.

The configurable models can be complete with other possible developments to adapt the requirements of other architecture modelling.

The modelling methods introduced in this work can be implemented to other projects, i.e. for other communication standards. A first integration in a wireless sensor network has been realized and presented with the SCNSL platform. It will be interesting to evaluate the implementation of our models other platforms like OMNet++ or WSNet.

Publications

Conférences internationales

F. Li, E. Dekneuvel, G. Jacquemod, D. Quaglia, M. Lora, François Pecheux & R. Butaud, “Multi-Level Modeling of Wireless Embedded Systems”, *FDL*, Munich, Germany, 2014

F. Li, E. Dekneuvel, R. Butaud, F. Pecheux & G. Jacquemod, “High-level modeling of Bluetooth system using Systemc-AMS”, *SAME*, University Booth, Sophia Antipolis, France, 2014

F. Li, R. Butaud, E. Dekneuvel & G. Jacquemod, "Bluetooth Transceiver Modeling Using SystemC-AMS", *IEEE PRIME*, Villach, Austria, 2013

Conférences nationales

F.Li, R. Butaud, E. Dekneuvel, G. Jacquemod & F. Pecheux, «Modélisation SystemC-AMS d’un Emetteur-Récepteur Bluetooth», *JNRDM*, Grenoble, France, 2013

Distinctions

Le projet CoCoE, dans lequel ce travail de thèse a été réalisé, a obtenu le **Trophée de la Recherche Publique Energie-Environnement-Climat** lors du *World Efficiency Congress*, en octobre 2015, Paris, France : http://atee.fr/energie-plus-magazine/actualites/mar-13102015-1523-world-efficiency-commence-avec-un-festival-de

References

[1] « Bluetooth Low Energy Technology | Bluetooth Technology Website » http://www.bluetooth.com/Pages/low-energy-tech-info.aspx

[2] « THIS IS ANT - the Wireless Sensor Network Solution », http://www.thisisant.com

[3] « ZigBee Alliance Home Page » [En ligne], http://www.zigbee.org/Home.aspx

[4] C. Jacquemod, C. Meyer, K. Sevin, L. Hebrard, K. Aguir, B. Nicolle, P. Lorenzini & G. Jacquemod, "Design and characterization of miniaturied remote current sensors", *ICSS,* Phuket, Thailland, Invited paper, pp. 78-79, 2015

[5] Kien Trung Nguyen, « Conception et réalisation d'un système de gestion intelligente de la consommation électrique domestique », PhD thesis report, UNS France, Dec. 2015

[6] Lydi Smaini, « RF analog impairments modeling for communication systems simulation: Application to OFDM-based transceivers », John Wiley & Sons (1st Edition), 2012

[7] G. Jacquemod, « TrustMe-ViP: Trusted Personal Devices Virtual Prototyping », EDA Tech Forum Journal, March 2009, p. 34-40

[8] H. De Man, F. Catthoor, G. Goossens, J. Vanhoof, J. Van Meerbergen, S. Note & J. Huisken, « Architecture-driven Synthesis Techniques for VLSI Implementation of DSP Algorithms », Proc. IEEE, Vol. 78, no.2, 1990, pp. 319-334

[9] Y. Lahbib, R. Kamdem, M.L. Benalycherif & R. Tourki, « An automatic ABV methodology enabling PSL assertions across SLD flow for SOCs modelled in SystemC », Sciencedirect.com, Computers and Electrical Engineering 31 (2005), pp. 282-302, Elsevier

[10] Thomas H. Lee, « The Design of CMOS Radio-Frequency Integrated Circuits », Cambridge University Press (2nd Edition), Dec. 2003

[11] Behzad Razavi, « RF Microelectronics », Prentice Hall (2nd Edition), 2012

[12] Michel Vasilevski, « Environnement de Conception Multi-Niveaux Unifiée Appliqué aux systèmes Mixtes », PhD thesis report, UPMC France, Sep. 2012

[13] Lucas Alves Da Silva, « Estimation de la consommation d'énergie et du taux d'erreur bit d'un système Bluetooth LE basée sur la modélisation hiérarchique d'un amplificateur de puissance à 2,4 GHz », PhD thesis report, UNS France, Nov. 2011

[14] « Bluetooth Core Specification v4.0 », http://www.bluetooth.com/Pages/Bluetooth-Smart.aspx

[15] Naresh Gupta , « Inside Bluetooth Low Energy », Artech House, 2013

[16] OSCI and IEEE, « IEEE Std 1666 - 2005 IEEE Standard SystemC Language Reference Manual », IEEE Std 1666-2005, pp. 1–423, 2006

[17] M. Barnasconi, C. Grimm, M. Damm, K. Einwich, M.M Louerat, T. Maehne, F. Pecheux & A. Vachoux, « SystemC-AMS extensions user's guide», Accellera Systems Initiative, 2010

[18] G. Jacquemod, L. Geynet, B. Nicolle, E. de Foucauld, W. Tatinian & P. Vincent, « Design and Modelling of a Multistandard Fractional PLL in CMOS/SOI Technology », Microelectronics Journal, vol. 39, n° 9, 2008, p. 1130-1139

[19] B. Nicolle, « Optimisation et validation de spécifications pour transmetteurs multistandards », PhD thesis report, UNS France, Sep. 2008

[20] M. Zorzi, N. Speciale, G. Masetti, « A new VHDL-AMS simulation framework in Matlab », BMAS, 2002

[21] Alexandre Lewicki, « Conception de Modèles Haut Niveau pour l'Optimisation et la Vérification de Systèmes Bluetooth», PhD thesis report, UNS France, Dec. 2008

[22] L. Alves Da Silva, A. Lewicki, B. Nicolle, E. Dekneuvel & G. Jacquemod, « Virtual RF System Platform Dedicated to Heterogeneous Complex SoC Design », Microelectronics Journal, vol. 43, n° 2, 2012, p. 98-109

[23] N. Bombieri, N. Deganello & F. Fummi, « Integrating RTL IPs into TLM designs through automatic transactor generation », in Proceedings of the conference on Design, automation and test in Europe. ACM, 2008, pp. 15–20

[24] Riviera Waves, « BT40 Modem Rx Front-End Algortihm Description », Internal report RW, version 1.0, 2010

[25] Riviera Waves, « BT40 Modem Tx Algorithm Description », Internal report RW, version 1.0, 2010

[26] John G. Proakis, « Digital Communications », 4th edition, Mc Graw Hill, 2000

[27] K.W. Martin, « Complex signal processing is not complex », IEEE Trans. on Circuits and Systems I: Regular Papers, Vol. 51, Issue 9, 2004, p. 1823 – 1836

[28] M. Vasilevski, N. Beilleau, H. Aboushady & F. Pecheux, « Efficient and Refined Modeling of Wireless Sensor Network Nodes Using SystemC-AMS », IEEE PRIME, Istanbul, June 2008

[29] H. Schmid, « How to use the FFT and Matlab's pwelch function for signal and noise simulations and measurements », August 2012, http://www.fhnw.ch/technik/ime/publikationen/2012/how-to-use-the-fft-and-matlab2019s-pwelch-function-for-signal-and-noise-simulations-and-measurements

[30] F. Li, E. Dekneuvel, G. Jacquemod, D. Quaglia, M. Lora, F. Pecheux & R. Butaud, « Multi-level modeling of wireless embedded system », FDL, Munich, 2014

[31] M. Conti & D. Moretti, « System level analysis of the Bluetooth », IEEE Conf. on Design, Automation and Test in Europe (DATE), Munich, 2005, p. 118–123

[32] S. Mc Canne & S. Floyd, « NS Network Simulator – version 2 », http://www.isi.edu/nsnam/ns

[33] A. Varga, « The OMNeT++ discrete event simulation system », ESM: European Simulation Multiconference, Prague, 2001

[34] C. Zhu, O. Yang, J. Aweya, M. Ouellette & D. Montuno, « A comparison of active queue management algorithms using the opnet modeler », IEEE Communications Magazine, vol. 40, n° 6, June 2002, p. 158–167

[35] « SystemC Network Simulation Library - Version 2 », 2013, http://sourceforge.net/projects/scnsl

[36] F. Herrera & E. Villar, « A framework for heterogeneous specification and design of electronic embedded systems in SystemC », ACM Transactions on Design Automation of Electronic Systems (TODAES), vol. 12, n° 3, 2007, p. 22

Annexe : Résumé étendu en français

Introduction

1. Motivation

Depuis plusieurs décennies, les objets communicants ont envahi notre quotidien et le nombre de données transmises, associées à un nombre croissant de standards de communication a explosé. Aujourd'hui les objets sont connectés entre eux, sans parfois d'intervention humaine, on parle alors de l'internet des objets (IoT : Internet of Things). Dans nombre de ces applications, ces objets doivent être autonomes en énergie et très bas coût. Pour la transmission de données bas débit, comme dans le cas de réseaux de capteurs, les standards de communication visent une très faible consommation comme par exemple BlueTooth Low Engergy (BLE), ANT+ ou zigbee.

Pour être compétitifs, les petits objets communicants doivent avoir un coût de développement et de production le plus faible possible. Cet objectif passe par l'étude et l'intégration dans un même environnement de simulation de différents flots de conception de circuits et systèmes ainsi que le développement d'une bibliothèque de modèles hiérarchiques d'IP-AMS. Il est important de prendre en compte les paramètres de conception critiques (estimation de la consommation, de la surface, détermination du BER, effets du canal, …) au niveau système afin de déterminer très vite les chemins critiques et pouvoir resserrer ou relâcher certaines contraintes.

Ce travail de thèse s'inscrit dans cet objectif de créer une plateforme virtuelle de conception et simulation de systèmes de communications sans fil dans un environnement unique multi-moteurs, associé à une bibliothèque hiérarchique de modèles comportementaux. Nous avons illustré cette approche dans le domaine des réseaux de capteurs sans fil utilisant le standard BLE pour des applications de bâtiments intelligents.

2. Contexte

Ce travail de thèse a été réalisé dans le cadre du projet CoCoE (Contrôle de la Consommation Electrique dans les bâtiments), labellisé par ARCSIS (Pôle de compétitivité Solutions Communicantes Sécurisées) et supporté par la plateforme Conception CIM-PACA. L'objectif final de ce projet vise l'émergence d'une solution innovante, non-intrusive et communicante de mesure détaillée des consommations électriques dans les bâtiments, par type et usage. Le développement de ce projet repose sur la technologie NIALM (Non Intrusive Application Load Monitoring), son implémentation sur FPGA et le développement de nouveaux capteurs sans fil. L'objectif de mon travail de thèse, en collaboration avec la société Riviera Waves (convention Cifre), est de modéliser et optimiser un transceiver BLE en utilisant le langage SystemC-AMS. Le standard BLE sera utilisé pour interconnecter les capteurs sans fil afin de monitorer la consommation électrique d'un bâtiment. Il convient de noter que le projet CoCoE a été récompensé lors du « World Efficiency Congress » (Paris, octobre 2015) en étant Laureat du Trophée de la Recherche Publique Energie-Climat-Environnement, attribué par l'ADEME et les deux magazines « EnergiePlus » et « Mesures ».

3. Organisation du mémoire

Ce mémoire de thèse, en plus d'une introduction et d'une conclusion générale, comprend trois chapitres :

- Chapitre 1 : Modélisation AMS pour la simulation de communications BT. Ce chapitre présente les méthodologies de description des systèmes hétérogènes : approche descendante ou Top-Down, approche montante ou Bottom-Up et approche mixte ou meet-in-the-middle.

Puis nous rappelons les différentes architectures de frontaux RF, ainsi que les différentes caractéristiques des segments RF et leur modélisation en Bande de Base (BB). L'accent est ensuite mis sur le standard BT. Pour conclure ce chapitre, nous montrons comment les langages SystemC et SystemC-AMS peuvent décrire les différentes parties du transceiver BLE.

- Chapitre 2 : Modélisation d'un transceiver BLE. Dans ce chapitre, nous décrivons les différents blocs de base d'un transceiver BT. Nous rappelons le principe de la modulation GFSK afin de décrire le bloc de génération de bits. Une architecture Zero-IF en émission et Low-IF en réception a été choisie. Les blocs de base « idéaux » sont décrits d'un point de vue fonctionnelle et validés par des simulations (intégrateur, corrélateur, modulateur, filtres, convertisseur, … : dans les différents domaines : numériques, analogiques et RF). Devant le temps de simulation prohibitif, les blocs RF sont réécrits en utilisant la méthodologie BB, puis raffinés afin de prendre en compte les non-idéalités qui vont impacter le couple consommation, BER. Chaque circuit (chaque modèle) est vérifié séparément, puis une première simulation système (point à point entre un émetteur et un récepteur) est effectuée. Enfin le BER est estimé. L'ensemble est fonctionnel, le temps de simulation acceptable et les résultats ont été validés à partir de mesure sur un circuit de la société Riviera Waves.
- Chapitre 3 : Modélisation et simulation Système. Ce chapitre est dédié à la simulation de cas réels (use cases) fournis par la société Riviera Waves afin de vérifier le fonctionnement de l'ensemble et d'estimer la consommation du système dans ces cas d'étude. Finalement, deux versions d'une même architecture sont modélisées, simulées et comparées. Les résultats montrent que la nouvelle version permet d'augmenter les performances globales du système, à la fois en terme de BER et de consommation. Pour conclure ce chapitre, nous avons montré qu'il était possible de réutiliser nos modèles pour une simulation très haut niveau (réseau de capteurs) en les intégrant dans la plateforme SCNSL, développée par l'université de Vérone, avec qui nous avons collaboré.

Chapitre 1 : Modélisation AMS pour la simulation de communications BT

1. Introduction

La conception de systèmes de communication sans fil est de plus en plus complexe en raison du nombre toujours croissant de fonctions embarquées et des contraintes liées à la consommation d'énergie, à la qualité de service et à la fiabilité requise. La vérification de ces différentes spécifications, fonctionnelles ou non, par des simulations est devenue incontournable. La simulation au niveau système est cruciale pour la vérification des systèmes de communication sans fil.

Selon les trois différents types de signaux (mixte –numérique et analogique- et RF), un système sans fil complet peut être divisé en trois domaines différents: le traitement numérique du signal (DSP), analogique en bande de base (BB) et radio fréquence (RF), comme le décrit la Figure 1.1.

Généralement, ces différentes parties sont décrites et simulées en utilisant respectivement des modèles de calcul (ou langages) et des outils différents. Ceci peut induire des incompatibilités, voire des erreurs, pour simuler le système complet. Il existe, en effet, un « gap » important pour simuler simultanément le niveau système (numérique), les signaux analogiques en bande de base et les porteuses RF. Par conséquent, nous avons besoin de techniques particulières pour gérer cette hétérogénéité et réaliser des simulations en cherchant à optimiser le couple précision des résultats et temps de simulation.

2. Méthodologie

2.1. Méthodologie de conception de circuits mixtes

La méthodologie retenue, nécessite l'étude et l'intégration dans un même environnement de simulation de différents flots de conception de circuits et systèmes ainsi que le développement d'une bibliothèque de modèles hiérarchiques d'IP-AMS. Dans le cas particulier des petits objets communicants, il est important de prendre en compte les paramètres de conception critiques (estimation de la consommation, détermination du BER, effets du canal, de la modulation, …) au niveau système afin de déterminer très vite les chemins critiques et pouvoir resserrer ou relâcher certaines contraintes. Pour atteindre cet objectif, nous avons introduit quatre niveaux hiérarchiques de modèles (cf. Fig. 1.2) :

- transistor (Spice or physical level)
- structurel (low level)
- comportemental (behavioral level)
- système (ESL or specifications level).

Pour développer ces modèles, nous utilisons les trois approches classiques de conception : bottom-up, top-down et meet-in-the-middle (MIM).

L'approche « Top-Down » est basée sur une conception qui commence avec les spécifications de haut niveau. Elle est généralement utilisé pour l'analyse de haut niveau et basée sur des simulations reposant sur des modèles logiques et/ou des modèles comportementaux qui prennent en compte les spécifications du système.

L'approche « Bottom-Up » est considérée comme une exploration de bibliothèques existantes contenant des modèles physiques de briques de base (building blocks) afin de construire une architecture permettant de satisfaire toutes les exigences et qui garantit la faisabilité du système. Mais la complexité des modèles augmente le temps de simulation, ce qui ne convient pas ou peu à l'exploration d'architectures complexes.

L'approche MIM est basée sur l'utilisation de modèles de niveau inférieur. Elle vise à extraire des performances plus précises de la conception de niveau inférieur pour ensuite enrichir la simulation de niveau supérieur par raffinement de ces modèles. Les paramètres génériques des modèles de haut niveau sont extraits de simulations bas niveau (ou de mesures). Le résultat de cette opération est que nous pouvons obtenir des résultats de simulation précis afin de vérifier les performances et les spécifications du système sans avoir à exécuter à nouveau des simulations bas niveau, certes précises mais prohibitives en temps de calcul.

La Figure 1.3 présente un exemple de cette approche MIM pour l'écriture d'un modèle simplifié d'interconnexion des systèmes ouverts (OSI). Le niveau physique est le plus bas niveau dans cette architecture, il comprend le MODEM et l'émetteur-récepteur RF. Les performances généralement extraites à partir de ce niveau sont le taux d'erreur par bit (BER) et la consommation d'énergie. Mais pour être capable d'obtenir un BER en prenant en compte les considérations de composants analogiques/RF, nous devons extraire les spécifications des circuits de bas niveau comme le gain linéaire, le facteur de bruit (NF), les non-linéarités et la puissance consommée par chaque bloc en fonction de son état.

Les performances extraites de chaque niveau d'abstraction peuvent être utilisées plus tard pour les modèles de niveau supérieur pour accélérer la simulation globale. Nous pouvons utiliser les spécifications extraites des différents niveaux inférieurs pour enrichir les différents niveaux plus élevés en les utilisant comme paramètres génériques.

La différence de fréquence entre le signal analogique en Bande de Base (liée au débit binaire) et la porteuse (RF) rend le temps de simulation très long et il n'est pas toujours possible de faire une co-simulation avec le niveau Datalink ou les niveaux plus élevés du système.

La limitation principale de cette méthodologie réside dans le développement de la bibliothèque hiérarchique. L'élaboration des modèles est difficile, et nécessite une expertise indéniable. Dans la conception RF, les exigences de l'utilisation de modèles exclusifs pour un type de simulateur limitent les possibilités de leur réutilisation dans différents environnements. Les simulateurs unifiés existants, associés à des langages tels que SystemC et SystemC-AMS, et les solutions de co-simulation de circuits mixtes permettent néanmoins d'atteindre cet objectif. En outre, chaque fois que ce sera possible, il convient de réutiliser des modèles existants et d'unifier différentes plateformes de façon transparente pour l'utilisateur. Les facteurs clés de succès d'une telle méthodologie sont les suivants:

- la capacité d'utiliser une approche hybride pour intégrer différents outils;
- l'utilisation de bases de données ouvertes en utilisant des langages standardisés;
- la disponibilité et la compatibilité des formats d'entrée et de sortie, ainsi que des interfaces standardisées.

Dans ce travail, nous utiliserons les méthodes MIM et de modélisation équivalente en bande de base, qui sera présentée au paragraphe suivant, pour réaliser une simulation du système global offrant une précision suffisante avec un temps de simulation acceptable.

2.2. Modélisation équivalente en Bande de Base

La méthode traditionnelle de sur-échantillonnage pour les simulations RF exige une fréquence d'échantillonnage très élevée pour conserver la précision du signal RF, engendrant des temps de simulation trop importants voire rédhibitoires lorsque l'on veut simuler plusieurs trames. Pour les standards de communications à bande étroite comme BT/BLE, puisque l'information utile intra-bande occupe un spectre très étroit, cette méthode gâche la plupart des efforts sur la représentation de l'information hors bande. La méthode de modélisation équivalente en BB propose de supprimer la porteuse afin d'accélérer la vitesse de simulation sans perdre le comportement analogique/RF, comme le montre la figure 1.4. Cette méthode exige que le signal RF soit à bande étroite et que le modèle BB équivalent puisse encore être en mesure de prendre en compte les non-idéalités du segment RF.

2.3. Architecture du frontal analogique/RF

Les architectures d'un récepteur peuvent être divisés en fonction des parties numériques, analogiques et RF avec l'utilisation d'un convertisseur analogique-numérique (ADC). La partie analogique/RF du récepteur est utilisée pour recevoir le signal RF et le convertir en bande de base. Il y a plusieurs architectures principales de récepteurs RF comme le récepteur hétérodyne, le récepteur homodyne (ou zéro-IF) et le récepteur pseudo zéro-IF (ou low-IF). Ces différentes topologies de récepteurs sont rappelées aux paragraphes 2.3.1 à 2.3.3 (cf. Figures 1.5 à 1.7).

2.4. Modélisation des blocs analogiques/RF avec l'approche MIM

Pour construire les modèles des circuits RF au niveau système, nous devons extraire leurs caractéristiques essentielles (à savoir celles qui impactent le couple BER, consommation), à partir de modèles bas niveau ou de mesures sur des circuits réels. Ces caractéristiques seront utilisées comme paramètres génériques des modèles « haut niveau ».

2.4.1. Modélisation du bruit

Dans la théorie des systèmes de communication, le bruit est généralement supposé additif, blanc et gaussien (AWGN, cf. Fig. 1.8). AWGN est normalement utilisé pour décrire des bruits ayant une distribution gaussienne comme le bruit thermique, présenté sur la figure 1.9. Mais dans ce travail, nous utilisons une source AWGN pour représenter le bruit général d'un circuit analogique et/ou RF. Ce bruit est une abstraction de toutes sortes de bruits internes du circuit en adaptant la puissance moyenne de bruit dans la bande considérée.

L'impact du bruit thermique dans les systèmes de communication est quantifié par la figure de bruit qui est définie comme le rapport entre le SNR (Signal to Noise Ratio) à l'entrée et le SNR à la sortie d'un dispositif. Le bruit d'un système en cascade est analysé avec l'équation de Friis, comme le montre la figure 1.10. La sensibilité du récepteur correspond au niveau de signal minimum à l'entrée

pour un SNR de sortie spécifié. Elle peut être facilement extraite de l'exigence de SNR minimum en présence de bruit thermique intégrée sur la largeur de bande du système.

Habituellement, les outils de simulation génèrent un bruit AWGN en utilisant une fonction renvoyant des nombres pseudo-aléatoires suivant une distribution normale avec une valeur efficace égale à 1. Dans le cas pratique de nos simulations, le niveau de ce bruit doit être ajusté à la valeur réelle du bruit thermique en tenant compte de la largeur de bande de simulation et de la fréquence d'échantillonnage f_s, qui est généralement beaucoup plus élevée que la bande passante utile, B, du signal (cf. Fig. 1.11). Dans des simulations précises, l'AWGN est réparti sur toute la bande de Nyquist $\pm f_s/2$; par conséquent, si l'on considère la valeur RMS de tension de bruit v_n intégrée sur une bande passante B, nous devons générer une tension de bruit RMS égale à $v_{n-simu} = v_n\sqrt{\frac{f_s}{2B}}$.

2.4.2. Bruit de phase

Le bruit de phase $\theta(t)$ est généralement décrit dans le domaine fréquentiel par sa DSP (Densité Spectrale de Puissance ou PSD en anglais) en dBc/Hz. Comme le montre la figure 1.12, il est le rapport entre la puissance de bruit mesurée sur une largeur de bande de 1Hz à un décalage fréquentiel f_Δ de la fréquence centrale et la puissance à cette porteuse.

Dans la conception analogique et RF, les performances des PLL sont généralement définies par leur profil de bruit de phase en sortie. Dans ce travail, nous modéliserons un bruit de phase pour un PLL complète dans le domaine de fréquence en utilisant le modèle présenté par l'équation 15. Ce modèle de bruit de phase peut être considéré comme un filtre passe-bas pour la modélisation BB ou un filtre passe-bande pour la modélisation RF en décalant les zéros de f_Δ vers f_c. Nous avons seulement besoin de convertir ce modèle en une source d'erreur de phase $\theta(t)$ en utilisant la transformée de Fourier inverse (IFT) avant de l'ajouter à l'oscillateur local (cf. Fig. 1.14).

2.4.3. Modélisation des non-linéarités

Les non-linéarités d'un circuit RF sont à l'origine de la compression du gain, la modulation croisée, et l'intermodulation. Il y a plusieurs façons pour prendre en compte ces non-linéarités. La figure 1.15 illustre le point de compression à 1dB. Dans ce travail, nous utiliserons principalement l'équation : $20\log A_{IIP3} = \frac{\Delta P}{2} + 20\log A_{in1}$, à savoir le coefficient d'ordre 3, illustré par le point d'interception d'ordre 3 en entrée (IIP3), pour évaluer la non-linéarité. L'analyse IIP3 en cascade est réalisé en utilisant l'équation en chaîne : $\frac{1}{A_{IP3}^2} \approx \frac{1}{A_{IP3,1}^2} + \frac{\alpha_1^2}{A_{IP3,2}^2} + \frac{\alpha_1^2\beta_1^2}{A_{IP3,3}^2} + \cdots$. Les figures 1.16 à 1.21 illustrent ce paramètre, ainsi que quelques conséquences. La figure 1.22 montre comment nous utiliserons la méthode de modélisation équivalente en BB pour prendre en compte ces non-linéarités d'ordre 3.

2.4.4. Puissance moyenne

La consommation moyenne est considérée comme un paramètre générique dans ce travail. Elle est extraite de la simulation SPICE ou de la mesure basée sur un circuit existant. Elle sera un paramètre important pour réaliser une estimation précise de l'énergie lors de la simulation haut niveau globale du système.

3. Standard Bluetooth Low Energy

3.1. Introduction

Dans ce paragraphe, nous présentons succinctement le standard BlueTooth (BT) et sa version faible consommation, Bluetooth Low Energy (BLE). Les principales architectures sont rappelées et comparées. L'accent sera mis sur la couche physique, PHY, avec les types de modulations utilisées.

3.2. BlueTooth et BLE

BlueTooth (BT) est une technologie de réseau personnel sans fil, géré par Bluetooth Special Interest Group (SIG) et a été définie comme une norme IEEE depuis sa première version ratifiée en 2002. Bluetooth Low Energy (BLE) a été complété dans le cadre de la spécification de Core BT version 4.0 en 2010 par Bluetooth SIG. Il est destiné à des communications bas débit peu énergivore

avec des paquets courts de données, une connexion rapide, un temps d'inactivité maximisé et un faible pic de puissance en émission ou en réception. BLE est largement utilisé par de nombreux dispositifs dans les domaines du sport et du bien-être, de la santé, des périphériques (claviers, souris, écouteurs), des balises et des appareils de divertissement. Dans le cadre du projet CoCoE, nous nous proposons d'utiliser ce standard pour développer un réseau de capteurs afin de monitorer la consommation électrique dans les bâtiments.

3.3. Architecture d'un système BLE

L'architecture BLE comprend deux types de dispositifs appelés respectivement monomodes et bi-mode Bluetooth v4.0. Le dispositif BLE monomode ne contient que la pile protocolaire BLE (cf. Fig. 1.23) qui peut communiquer avec d'autres appareils BLE, ainsi qu'avec des appareils BLE bi-mode quand ils utilisent la partie de la technologie BLE de leur architecture. Les dispositifs BLE bi-mode sont compatibles avec la pile classique BT et la pile BLE. Cela les rend capables de communiquer à la fois avec des dispositifs BT classiques et des appareils BLE.

Bien que notre travail soit concentré sur la modélisation de la couche PHY de l'émetteur-récepteur, il est très important de comprendre également les fonctionnalités et le comportement de la partie LL (LinkLayer) pour la simulation finale d'un système BLE complet. En effet, le modèle au niveau LL sera raffiné par des extractions à partir des modèles PHY de l'émetteur-récepteur. La modélisation de l'interface entre la couche PHY de l'émetteur-récepteur et sa couche LL est ainsi absolument nécessaire.

3.4. Couche LL de BLE

La couche LL d'un appareil BLE est placée entre la couche PHY et la couche L2CAP. Elle est en charge du contrôle, de la gestion, de l'établissement des liens, de la sélection de fréquences pour transmettre des données, en supportant différentes topologies et diverses manières pour échanger ces données. La fonctionnalité principale de LL contient : Préambule, Adresse d'accès, protocole des trames, génération et vérification du CRC, blanchiment des données (data whitening), génération de nombres aléatoires et cryptage AES.

3.4.1. Couche LL de BLE

La machine d'états, présentée sur la figure 1.24, décrit le fonctionnement de la couche LL, avec les cinq états : Standby, Advertising, Scanning, Initiating et Connection. La description de chaque état est résumée dans le Tableau 1.2.

3.4.2. Format d'un paquet BLE

Le LL utilise un format de paquet unique qui contient le Préambule, l'adresse d'accès (AA), le PDU et le CRC, comme le montre la figure 1.25. Le préambule, codé sur 8 bits, est utilisé par le récepteur pour effectuer la synchronisation de la fréquence et l'estimation du timing des symboles. Il vaut 10101010b pour les paquets du canal Advertising, et pour les paquets du canal de données. Il vaut 10101010b si le LSB de AA est égal à 0 et 01010101b si son LSB est égal à 1. Pour tous les paquets du canal Advertising, AA doit être égal à 10001110100010011011111011010110b (soit 0x8E89BED6 en hexadécimal). Dans les paquets du canal de données, il est généré avec des restrictions dans la version BT Core4 et utilisé pour une demande de connexion. Le format de PDU dépend de la nature (Advertising ou données) du paquet.

Data whitening est utilisé pour éviter les longues séquences de "1" ou de "0". Une politique de filtrage au niveau du LL permet d'économiser l'énergie, en utilisant une «liste blanche» qui est configurée par le dispositif maître pour filtrer les advertisers, les scanners et les initiateurs non-désirés.

3.4.3. Opérations de la machine d'états LL

Quand le LL est dans un état « non-Standby », il va échanger des PDUs qui correspondent aux différents événements suivants :

- **Advertising & Scanning:** Lorsque le LL est dans l'état Advertising, il doit utiliser les canaux advertising pour envoyer et recevoir des PDUs suivant 4 types d'événements advertising comme indiqués dans le Tableau 1.3, extrait de la spécification BT 4.0. Lorsque LL est dans l'état de scanning, il peut effectuer soit un scanning passif où il ne peut que écouter sur un certain canal advertising et recevoir des paquets mais sans en envoyer, ou un scanning actif où il peut demander des informations supplémentaires auprès de l'advertiser.

- **Advertising & Initiating:** Lorsque LL est dans l'état initiating, il ouvre une ScanWindow pour écouter sur chaque canal advertising et communiquer avec les advertisers. Il peut demander une connexion et entrer dans l'état de Connection.
- **Connections:** un LL qui entre dans l'état de Connection de l'état Initiating devient un Maître. Inversement, quand il entre de l'état Advertising, il devient un Esclave. Il est autorisé à communiquer avec les PDUs du canal de données. Une connexion est créée quand un LL entre dans cet état. Le Maître et l'Esclave utilisent un timer de supervision pour détecter une perte de connexion possible à tout moment. Une supervision « timeout » sera créée dans le cas d'un timeout, et fera sortir le LL de l'état Connection pour revenir à l'état Standby. Pendant l'échange de paquets dans un événement de connexion, le Maître et l'Escale indiquent s'ils ont plus de données à envoyer pour décider s'ils maintiennent toujours l'événement de connexion ou non.

Ces mécanismes sont illustrés par les figures 1.26 à 1.28.

3.5. Description de la couche Physique de BLE

La couche physique (PHY) de BLE définit la façon dont les signaux sont modulés, démodulés et transmis par l'émetteur-récepteur. La radio BLE fonctionne dans la bande ISM 2.4 GHz. La gamme complète de fréquences utilisées par BLE est de 2,400 à 2,4835 GHz. Comme le montre la figure 1.29, elle est divisé en 40 canaux qui ont des fréquences centrales données par la relation 2402+k×2 MHz, où k varie de 0 à 39. Les trois canaux avec les indexes 37, 38 et 39 sont utilisés comme canaux advertising pour échanger des données de diffusion et établir des connexions BLE. Les 37 autres canaux sont les canaux de données qui sont utilisés pour échanger des données pendant les connexions. BLE utilise la technique de saut de fréquences (FH) pour lutter contre les interférences et les évanouissements par commutation rapide de l'indexe du canal suivant une séquence pseudo-aléatoire.

3.6. Modulation GFSK – Modulation directe ou indirecte

Le standard BLE utilise la modulation GFSK avec un débit de 1 Mbit/s. GFSK utilise un filtre gaussien pour lisser chaque symbole numérique avant la procédure de modulation de fréquence (cf. Fig. 1.30), pour éviter les hautes fréquences dues à la commutation et pour réduire ainsi la largeur de bande spectrale du signal. Cette modulation peut être effectuée de manière directe ou indirecte, comme le montre la figure 1.31. L'indice de modulation doit être compris entre 0,45 à 0,55. Il est défini pour une puissance de sortie de l'émetteur entre 0.01mW à 10mW. Le niveau de sensibilité du récepteur est réglée pour être inférieure ou égale à -70 dBm, ce qui correspond à un BER de 0,1%.

4. Outils pour la simulation de systèmes mixtes

4.1. Introduction

Lors de la conception de systèmes de communication, il est nécessaire d'utiliser des outils et des langues de modélisation différents. Pour être efficaces, ces langages et outils doivent offrir des simulations précises dans des temps acceptables. Nous présentons, dans les paragraphes suivants, les outils permettant la description de systèmes hétérogènes (analogiques, mixtes et RF).

4.2. SystemC

SystemC est un ensemble de classes C++ et de macros pour la modélisation au niveau système, la conception et la vérification. Il a des similitudes sémantiques avec les langages de description matérielle tels que VHDL et Verilog. En outre, il permet de modéliser les composants logiciels en utilisant directement le langage C/C++ et en héritant de tous les avantages du C++ comme les constructions riches de langage et les types de données qui réduisent ainsi le temps de conception et les efforts de travail. SystemC supporte la conception de différents niveaux d'abstraction afin que les concepteurs puissent modéliser soit un système complexe à un niveau d'abstraction très élevé ou un bloc analogique bas niveau avec des primitives électriques prédéfinies.

SystemC-AMS est une extension de SystemC pour la modélisation des composants du signal analogique et mixte (AMS) avec trois modèles de calculs (ou MoC, Model of Computation) spécifiques comme le Timed Data Flow (TDF), le Linear Signal Flow (LSF) et l'Electrical Linear Network (ELN) pour adapter les exigences de la modélisation à différents niveaux d'abstraction.

Verilog-AMS (respectivement VHDL-AMS) est l'extension aux circuits analogiques et mixtes du langage VerilogHDL (respectivement VHDL). Il est utilisé pour modéliser des blocs par une description comportementale ou structurelle (en encapsulant d'autres sous-circuits avec leurs ports d'entrée-sortie et des paramètres génériques). Il simule les comportements analogiques avec une solution conservative en respectant les lois de Kirchhoff.

Matlab est un outil puissant est largement utilisé dans le monde. Développé par la société MathWorks, il est basé sur le langage C. Il est généralement utilisé pour les calculs matriciels, les fonctions de traçage, etc., et peut s'interfacer avec des programmes écrits en C, C ++, Java et Python. Appliqué pour la conception de systèmes embarqués, Matlab est également un simulateur de flux de données. Il est capable de modéliser des systèmes analogiques/RF contenant des non-idéalités et l'outil Simulink est facile et intuitif pour les descriptions de modèles. Toutefois, il est limité en vitesse de simulation, notamment pour les systèmes RF.

4.3. Interopérabilité entre les MoCs de SystemC

Les différents modèles de calcul de SystemC peuvent être utilisés pour mélanger différents modèles à différents niveaux d'abstraction : TDF, LSF, RTL et TLM MoCs.

SystemC présentent certaines restrictions: les processus d'événements discrets de SystemC ne sont pas autorisés dans les modules TDF, les composants de SystemC-AMS TDF et LSF ne peuvent pas être connectés directement aux composants TLM, …. Par conséquent, il est nécessaire de réaliser des mécanismes d'interface afin d'assurer cette interopérabilité. La figure 1.32 présente les mécanismes qui ont été développés, dans le cadre de ce travail de thèse, pour aboutir aux simulations finales de communications BLE à différents niveaux d'abstraction afin d'obtenir des résultats les plus précis possibles avec des temps relativement « courts ».

4.4. Conclusion

Dans ce travail, nous avons choisi d'utiliser les outils de SystemC pour modéliser un système de communications. En effet, ils correspondent aux exigences requises pour simuler un tel système entier. Ces outils prennent en charge les modèles de haut niveau qui peuvent être simulés rapidement, tout en étant raffinés par des extractions de plus bas niveau. En outre, ils couvrent tous les domaines (hétérogènes) d'un tel système de communication sans fil. Ils permettent, en effet, un interfaçage facile, rapide et efficace entre ces différents domaines.

5. Conclusion

Dans ce chapitre, nous avons présenté les challenges liés à la modélisation et la simulation d'un système de communication sans fil complet, contenant différents domaines (numérique, analogique et RF). Pour résoudre ces problèmes, nous avons choisi la méthode de modélisation dite MIM et les outils basés sur SystemC. Nous avons aussi rappelé les principales caractéristiques des blocs RF, afin de pouvoir les modéliser d'un point de vue fonctionnelle, puis par raffinement de ces modèles issus de simulation bas niveau ou de mesures sur des circuits déjà réalisés. La figure 1.32 montre quels langages seront utilisés en fonction des trois différents domaines (types de signaux) de l'émetteur-récepteur. Cette approche nous permet de réutiliser les véhicules et vecteurs de test (use case BLE) développés par la société Riviera Waves en utilisant SystemC-TLM. Le chapitre suivant est focalisé sur le développement des modèles, leur raffinement et leur validation en utilisant SystemC-AMS pour décrire un émetteur-récepteur BLE. Pour cela, nous utiliserons l'architecture d'un circuit développé par la société Riviera Waves.

Chapitre 2 : Modélisation d'un transceiver BLE

1. Introduction

Ce chapitre présente l'architecture du transceiver BLE, développée par la société Riviera Waves, que nous avons choisie comme véhicule de test. L'objectif final est de modéliser cet émetteur-récepteur au niveau système en raffinent suffisamment ces modèles afin de prendre en compte les spécificités des blocs RF pour effectuer des simulations système précises avec des temps acceptables. Cette simulation haut niveau doit permettre soit de faire des explorations d'architectures, soit de

valider et/ou optimiser une architecture en relâchant ou resserrant certaines contraintes au niveau de certains circuits (redéfinition des spécifications des blocs de base).

Un transceiver BLE complet peut être séparé en 4 blocs fonctionnels effectuant la modulation, la démodulation, l'émission et la réception (cf. Fig. 2.1). Dans ce travail, l'émetteur-récepteur contient le modulateur et le démodulateur GFSK. Pour la partie RF, un émetteur de type Zéro-IF et un récepteur de type Low-IF (fréquence intermédiaire) ont été choisis. La partie MODEM utilise la modulation GFSK et la modulation FSK est réalisée par la méthode indirecte (cf. Fig. 2.2). Les déviations de fréquence sont converties en phases instantanées et transmises comme un signal modulé en phase.

Notre étude est focalisée sur l'analyse et l'optimisation des performances Rx. Dans un premier temps, la modélisation des blocs Tx sera idéale. La fréquence intermédiaire de l'architecture Low-IF du récepteur RF est de -1 MHz. Dans cette chaine de traitement le signal est alors complexe. Le signal IF est filtré par un filtre passe-bande complexe et numérisé par un ADC ΔΣ passe-bande. La down-conversion vers la bande de base est traitée par un DSP (cf. Fig. 2.3). Dans la partie DSP, la démodulation est le processus inverse de la modulation. Le signal est ramené en bande de base, et $\Delta\varphi(t)$ est calculée puis convertie en $\Delta f(t)$ par la fonction tan (·).

2. Modélisation et simulation du transceiver

2.1. Modélisation du modulateur GFSK

2.1.1. Architecture du modulateur

Le processus de modulation GFSK, décrit par la figure 2.4, commence avec un générateur de bits qui envoie des "0" ou "1" à un débit binaire de 1 MHz. Ces bits sont signés par un bloc BFSK qui utilise deux symboles avec une différence de phase de 180° pour représenter les bits comme "-1" pour "0" et "+1" pour "1". L'étape suivante est le processus de modulation de fréquence indirecte. Les données signées en bande de base sont mises en forme par un filtre gaussien (GF) qui génère une sortie de 13 échantillons au cours de chaque période de données d'entrée, ce qui rend le débit de données de sortie à 13 MHz. La sortie du GF est considérée comme la déviation de la fréquence instantanée $\Delta f(t)$, qui est intégrée en tant que déviation de phase instantanée, $\Delta\varphi(t)$, par un intégrateur numérique. Ce signal de phase est ensuite « mappé » en tant que $e^{j\Delta\varphi(t)}$ en utilisant une transmission en quadrature : $\cos[\Delta\varphi(t)]$ pour la branche en phase (chemin I) et $\sin[\Delta\varphi(t)]$ pour la branche en quadrature (chemin Q). La dernière étape consiste à convertir le signal numérique en analogique avec le DAC. Puisque l'émetteur est considéré comme idéal, le modèle du DAC est simplifié par un convertisseur parfait qui ne génère aucun bruit en sortie.

2.1.2. Générateur de bits, filtre gaussien et intégrateur numérique

Chacun des blocs de base (cf. Fig. 2.4) sont alors détaillés et présentés respectivement sur les figures 2.5 à 2.8. On rappelle également les équations permettant de décrire le fonctionnement de chaque bloc et qui seront intégrées dans leurs modèles respectifs.

2.2. Modélisation du démodulateur

2.2.1. Architecture du démodulateur

La démodulation est réalisée en utilisant une série d'opérations inverses de celles de la procédure de modulation. Une opération "arctangente" est utilisée pour détecter la déviation de phase. Ce signal $\Delta\varphi(k)$ correspond à la sortie du bloc de "conversion fréquence/phase" dans le processus de modulation. Ensuite, une dérivation est utilisée pour convertir cette information en déviation de fréquence. A la fin, nous avons également besoin d'un corrélateur pour retrouver les bits émis initialement par le "Générateur de bits". La figure 2.9 illustre cette architecture.

2.2.2. Opération Arctangent et dérivation

Pour obtenir les informations de déviation de fréquence, nous avons besoin d'une dérivation de la phase. Toutefois, une correction est nécessaire pour corriger des sauts de phase de 2π (opération arctangent) dus à l'émission de longues séries de "0" ou de "1". Ce bloc correcteur remplace les valeurs erronées par les échantillons précédents.

2.2.3. Corrélateur

Le système a besoin d'un corrélateur pour détecter le code d'accès (AC) BT ou les adresses d'accès (AA) BLE. Il permet au système de déterminer l'instant où il doit commencer à échantillonner

le flux de données sur-échantillonné entrant afin de le sous-échantillonner et de récupérer le débit binaire initial de 1 MHz. Les résultats de corrélation dans un système BT/BLE est utilisé pour extraire les données de PDU dans un paquet reçu par la détection de son AC/AA avec le mot de synchronisation. La structure du corrélateur est donnée par la figure 2.10.

2.3. Modélisation fonctionnelle de l'émetteur

2.3.1. Architecture de l'émetteur

L'émetteur RF est composé d'une conversion numérique-analogique (DAC), d'un filtrage passe bas et d'une conversion RF (cf. Fig. 2.11). La modélisation du DAC peut être réalisée en répétant les échantillons numériques (fonction de maintien). Ce processus entraîne un repliement de spectre, ce qui nécessite des filtres passe-bas (LPF) pour éliminer les harmoniques avant la conversion du signal RF, par mélange avec les oscillateurs locaux.

2.3.2. DAC et LPF

Le DAC est modélisé comme un interpolateur dont le nombre de bits en sortie est fixe. Il duplique le même échantillon N fois, où N est la valeur de sur-échantillonnage. La fréquence de simulation est choisie à $f_s = 10{,}816\,GHz$. Cette interpolation, indispensable lors de la simulation, n'existe pas dans un vrai circuit. Elle permet de simuler le caractère temps continu des signaux analogiques. Il est recommandé de choisir un rapport d'interpolation adapté aux différentes fréquences.

Le filtre passe bas, LPF (cf. Fig. 2.12), est un filtre Butterworth du 3^{ième} ordre avec une bande passante de 1 MHz. Il est décrit par sa fonction de transfert H(s) (cf. Eq. 7), qui peut être modélisée par la Fonction de Transfert de Laplace (LTF) proposée par SCAMS (TDF MoC).

2.3.3. Conversion RF

La fréquence de l'oscillateur local dans l'émetteur est fixée à 2.402 GHz. L'oscillateur génère deux porteuses en quadrature : $\cos(\omega_c t)$ et $-\sin(\Delta\varphi)$, qui sont utilisés pour générer le signal RF final qui est envoyé par l'antenne en multipliant avec $\cos(\Delta\varphi)$ et $\sin(\Delta\varphi)$. La figure 2.13 illustre ce spectre de sortie, avec une fréquence centrale de 2,4021 GHz.

2.4. Modélisation fonctionnelle du récepteur

2.4.1. Architecture du récepteur

L'architecture du récepteur RF est de type Low-IF, dont la fréquence d'intermédiaire a été fixée à -1 MHz. Comme le montre la figure 2.14, ce récepteur contient les blocs suivants : un amplificateur faible bruit (LNA), un mélangeur, un filtre complexe passe-bande (CXF1) et un ADC de type ΣΔ. La fréquence de travail de cet ADC est de 52 MHz. La première version du récepteur RF sera une modélisation idéale fonctionnelle. La représentation idéale de LNA est un gain linéaire et pour le LO une paire de porteuses en quadrature.

2.4.2. Down-conversion à -1MHz

Pour obtenir le spectre du signal centré à -1 MHz, il faut déplacer le spectre du signal (centré sur f_c) vers la gauche en multipliant le signal par $e^{-j\omega_c' t}$, où $\omega_c' = 2\pi \times (f_c + 1\,MHz)$. Le spectre du signal décalé est illustré par la figure 2.15. Ce processus transforme le signal réel en un signal complexe, qui doit être filtré par le filtre complexe CXF1.

2.4.3. Modélisation du filtre complexe

Le filtre complexe (CXF1), placé après le mélangeur, est décrit par l'équation 9. Il existe deux façons pour réaliser la modélisation de sa fonction de transfert. La première façon est de construire un modèle architectural directement en décrivant sa topologie présentée sur les figures 2.16 et 2.17. Ce procédé est original et intuitif, mais il est difficile de réutiliser le modèle ainsi construit dans le cas d'une modification du bloc.

La deuxième façon consiste à utiliser les LTF proposées par SCAMS pour modéliser tout d'abord un LPF en décalant la fréquence centrale du CXF1 vers le continu. Le signal d'entrée est décalé aussi vers le continu. Puis après un filtrage passe bas, le signal est ramené en haute fréquence pour retourner à la fréquence centrale de CXF1 (cf. Fig. 2.18). Le modèle complexe obtenu en utilisant cette méthode peut être « encapsulé » avec des paramètres comme les zéros, les pôles et la fréquence centrale.

Les figures 2.19 et 2.20 représentent respectivement la fonction de transfert du filtre CXF1 et le spectre du signal en sortie.

2.4.4. ADC ΣΔ

L'ADC est un bloc complexe comme le CXF1, dont les voix I et Q sont croisées. La mise en œuvre de ce bloc est décrite par la figure 2.21, où tous les signaux (et coefficients) sont complexes. SCAMS n'intègre pas directement des filtres FIR numériques ; toutefois, nous pouvons facilement réaliser un filtre FIR avec la boucle "for" du C/C++. La figure 2.22 présente le spectre du signal en sortie de ce filtre.

2.5. Modélisation de la partie numérique en BB du récepteur

Dans la parte bande de base numérique (cf. Fig. 2.23), la sortie du convertisseur est d'abord filtrée par un autre filtre passe-bande complexe (CXF2) pour éliminer le bruit hors bande. Les blocs suivants sont deux filtres 4-tap identiques qui sont utilisés pour atténuer le bruit, cette fois dans la bande utile du signal. Après le filtrage, le débit de données est ramené de 52 MHz à 13 MHz par une décimation par 4, puis le signal est décalé de la fréquence intermédiaire en continu. Enfin, les bruits hors bande non-désirés sont éliminés par un LPF avant d'entrer dans le processus de démodulation.

Le filtre CXF2 est un filtre FIR complexe d'ordre 127 avec 128 coefficients complexes (cf. Fig 2.24). Ces coefficients sont générés par Matlab avec les paramètres comme l'ordre, le type de filtre, la fréquence d'échantillonnage, les fréquences de coupure et le réglage de la bande passante et la bande d'arrêt. Les figures 2.25 à 2.27 illustre les caractéristiques de ce filtre.

Le filtre 4-tap est un filtre FIR avec 4 coefficients. Il est utilisé pour atténuer le bruit dans la bande de fréquences utiles avec un sous-échantillonnage de 4. La figure 2.28 illustre les caractéristiques de ce filtre.

2.6. Simulation du transceiver à partir des modèles fonctionnels

La première version de la plate-forme d'émission-réception, modélisée de façon idéal et fonctionnel, est opérationnelle. Lors de la première simulation, nous avons généré en entrée un paquet BLE de 64 bits, afin de récupérer les bits en réception et estimé le temps de simulation. Le résultat montre que le système fonctionne correctement, car en sortie du corrélateur on retrouve le paquet émis sans erreur (cf. Tableau 2.1). Toutefois, le temps de simulation est trop long pour un seul paquet BLE, et sera prohibitif pour simuler un « use case » réaliste.

3. Optimisation du temps de simulation

Pour être capable de simuler un grand nombre de trames BLE (i.e. de paquets), nous utilisons la méthode de modélisation en BB équivalente, expliquée dans le chapitre 1. Nous choisissons comme fréquence de simulation de la partie RF, la valeur fs = 52 MHz, qui est la même fréquence de travail pour l'ADC ΣΔ dans le récepteur (cf. Fig. 2.32). Cette fréquence est la valeur la plus faible qui puisse être utilisée en tant que fréquence de simulation (pour cette plateforme) pour maintenir la performance du SNR du système en sortie de l'ADC. Puisque le spectre de la bande de simulation ($+/-\frac{f_s}{2}$) est la copie du spectre dans la bande, nous allons trouver le spectre du signal autour de +/- 10 MHz (au lieu de +/-2.402GHz), comme le montre la figure 2.31.

Ce modèle d'émetteur-récepteur est re-simulé avec le même paquet BLE. Puisque le SNR à la sortie de l'ADC n'est pas changé, l'émetteur-récepteur fonctionne correctement comme dans la première simulation, mais le temps de simulation est fortement réduit comme nous le montrerons dans la suite de ce chapitre (cf. paragraphe 5, tableau 2.11).

4. Raffinement des modèles en utilisant les caractéristiques du frontal RF

Les spécifications RF, y compris l'impédance d'entrée et de sortie, le gain linéaire, l'IIP3, la NF et la consommation d'énergie seront prises en compte dans la modélisation du récepteur pour évaluer ses performances à partir des simulations en BB équivalente (BBE).

Dans la simulation BBE, nous utilisons la technique décrite au chapitre 1. Nous avons ainsi introduit un type de donnée en C++ : BB (DC, I1, I2, I3, Q1, Q2, Q3) afin de prendre en compte les non-linéarités d'ordre 3.

4.1. Modélisation du canal

Le canal RF dans cette simulation est modélisé en prenant en entrée un signal dont la puissance est limitée par un dispositif de commande d'intensité de signal qui convertit la puissance du signal

désiré en dBm par un facteur d'atténuation. Dans l'analyse de la performance de la chaîne Rx, nous utilisons un ton unique comme l'entrée de Rx qui a une puissance moyenne fixe pour mesurer la performance en termes de SNR. Il est facile de contrôler la puissance de signal avec cette méthode, comme le montre la figure 2.33.

4.2. Méthode de raffinement pour la partie analogique du récepteur

Un bloc analogique/RF peut être décrit comme une série de spécifications communes à chacun avec laquelle nous pouvons « encapsuler » un bloc sans tenir compte de tous les détails de bas niveau. Les blocs analogiques principaux devant être raffinés sont le LNA, le mélangeur, le CXF1 et les LO.

En conclusion, chacun des modèles fonctionnels idéaux de ces blocs est modélisé par une certaine opération (gain pour le LNA, produit des entrées pour le mélangeur, …). La sortie de chaque bloc peut être considérée comme une fonction de l'entrée comme $out = F(in)$ en général (cf. Eq. 15).

Nous prendrons comme unité la tension (plutôt que le courant ou la puissance). Le bruit ajouté par un bloc total peut être modélisé comme une tension additive de bruit δ ramenée à l'entrée dont la valeur efficace est donnée par $\delta = \sqrt{4(10^{NF/10} - 1)KTR\frac{f_s}{2}}$. La non-linéarité peut être modélisée en considérant seulement l'onde fondamentale et l'harmonique d'ordre 3 du signal par la fonction $F_r(x) = \alpha_1 x - \alpha_3 x^3$ où x est l'entrée d'origine plus la densité de bruit ramenée à l'entrée (+ δ). α_1 est le gain linéaire du bloc et α_3 est le coefficient du 3ème harmonique calculé en utilisant la relation $A_{IIP3} = \sqrt{\frac{4}{3}\left|\frac{\alpha_1}{\alpha_3}\right|}$ qui a été introduit dans le chapitre 1. Cette méthodologie de raffinement peut être illustrée par la figure 2.34.

4.3. Raffinement des blocs analogiques/RF du récepteur

Dans les sous-paragraphes (4.3.1 à 4.3.4), nous présentons les raffinements apportés à ces trois blocs de base (LNA, mélangeur et CXF1). Les valeurs des spécifications bas niveau ont été fournies par la société Riviera Waves afin d'avoir des valeurs réalistes. Elles sont introduites dans les modèles sous forme de paramètres génériques. Leurs impacts respectifs sur le signal à chaque étape de la chaîne de traitement sont simulés et validés par les mesures. Les figures 2.35 à 2.38, ainsi que les tableaux 2.2 à 2.8 résument l'ensemble du processus suivis et les résultats de simulation et de validation des modèles.

4.4. Vérification des spécifications en cascade

Une fois les modèles analogiques/RF validés, il est possible de simuler et vérifier les paramètres du frontal RF complet, comme la NF et l'IIP3. La simulation NF de la chaîne est mise en œuvre avec la configuration donnée dans le Tableau 2.9, et celle l'IIP3 par la configuration du Tableau 2.10. Les résultats de la simulation (cf. Fig. 2.19) correspondent bien aux valeurs spécifiées.

5. Simulation du transceiver et estimation du BER

Pour estimer le BER de l'émetteur-récepteur, la chaîne Rx est configurée suivant les valeurs données dans le Tableau 2.11. La simulation est exécutée par l'envoi de 1000 paquets BLE, entrelacés par des paquets « vides ». Chaque trame BLE (ou paquet) contient 64 bits, soit au total 128000 bits. Le temps virtuel à simuler est donc de 128 ms.

Les signaux d'entrée ont des puissances variant de -93dBm à -91dBm et le plancher de bruit du récepteur est fixé à -114dBm. La performance en termes de BER obtenue et la sensibilité du Rx sont montrées sur la figure 2.42. D'après cette simulation, la sensibilité du Rx est de -95,3 dBm pour un BER de 0,1%. Le temps de simulation n'est cette fois que de 3,7 ms/bit (à comparer aux 300 ms/bit du Tableau 2.1, ce qui donne un « speed-up » de l'ordre de 80). Cette durée est acceptable et nous permet d'envisager des use cases réalistes pour la simulation globale du système BLE.

6. Conclusion

Dans ce chapitre, nous avons introduit la modélisation au niveau système en utilisant SCAMS pour un émetteur-récepteur BLE. La première version est un modèle fonctionnel idéal qui a été validé en exécutant une simulation à partir d'un seul paquet BLE. La plateforme de simulation est opérationnelle, toutefois le temps de simulation est trop élevé et devient rédhibitoire pour des

simulations ultérieures plus complexes. La méthode BBE a été appliquée afin de réduire le temps de simulation. En outre, les modèles ont été raffinés afin de prendre en compte les non-idéalités des blocs du transceiver afin de réaliser des simulations permettant d'estimer de manière réaliste les performances du système comme le BER. Cela a fonctionné correctement et le temps de simulation a été considérablement réduit.

Le BER du récepteur a finalement été estimé en exécutant les modèles raffinés avec 1000 paquets BLE. Le résultat obtenu est conforme aux spécifications et le temps de simulation est acceptable pour envisager la simulation de use cases fournis par notre partenaire industriel.

Dans le chapitre suivant, les blocs RF seront construits avec des modèles reconfigurables pour la réutilisation facile et rapide lors de simulations efficaces de systèmes BLE, ou pour des futures applications telles que l'analyse de la performance d'une chaîne Rx modifiée (spécifications de blocs ou de l'architecture de la chaîne). Le modèle d'émetteur-récepteur configurable sera intégré dans la plateforme BLE numérique en SCTLM avec une interface qui permettra de relier les deux domaines avec les signaux de contrôle RF à la partie numérique et la conversion de données.

Chapitre 3 : Modélisation et simulation système

1. Introduction

Dans ce chapitre, nous nous proposons d'utiliser les modèles développés précédemment pour simuler un système complet de communications BLE, notamment pour une approche réseau, en incluant des caractéristiques analogues/RF.

Dans un premier temps, nous avons réalisé un paramétrage automatique des différents modèles afin de pouvoir les modifier facilement et rapidement pour simuler efficacement différents « use cases ». De même, pour créer une « netlist » lors d'une simulation particulière d'un système global, nous sommes partis d'un modèle « commun » à tous les blocs (circuits) de base d'un émetteur-récepteur (ou transceiver) RF.

En outre, la partie numérique d'un dispositif BLE est modélisé en SC-TLM. Le plus bas niveau d'abstraction de cette partie correspond à la couche LL. Pour connecter l'émetteur-récepteur à cette couche LL, il faut remplacer le modèle original du "canal" par le modèle de l'émetteur-récepteur décrit en SC-AMS (cf. Fig. 3.1), et ainsi créer le lien de contrôle entre le domaine numérique et le domaine analogique. Les signaux de contrôles, y compris les signaux d'activation de l'émetteur-récepteur, sont utilisés pour estimer la consommation d'énergie de chacune de ses parties. Ce processus est très utile pour comprendre la relation entre la consommation d'énergie et les performances d'un transceiver lors d'une simulation donnée, pour évaluer par exemple une nouvelle conception.

2. Blocs paramétrables du récepteur

Pour effectuer des simulations globales, la première étape consiste à créer la netlist « haut niveau » pour décrire le système complet de communication qui sera simulé. Dans un souci de simplification et d'efficacité, cette netlist ne contient pas tous les détails de chaque instance. En effet, si ce nombre de composants (blocs de base instanciés) est élevé et que pour chaque modèle, le nombre de paramètres génériques soit également important, il conviendrait pour certaines simulations de modifier à chaque fois l'ensemble de ces paramètres avant de les re-tester. Cette opération fastidieuse, même automatisée, est risque d'erreur, et le temps de simulation risque de devenir trop grand. Nous avons ainsi essayé d'unifier les modèles de chaque bloc pour les niveaux d'abstraction élevés, en déterminant leurs caractéristiques communes.

Ainsi, les blocs RF peuvent être décrits par les mêmes équations qui prennent en compte leurs non-linéarités ou le bruit. Les différents filtres peuvent être représentés par leur fonction de transfert sous une forme unifiée. La construction d'un modèle commun pour les blocs de base, qui contiendra les caractéristiques nécessaires aux objectifs des « use cases », nous permettra de les réutiliser facilement et efficacement lors de simulations globales.

LNA paramétrable : Le LNA est le bloc RF le plus basique qui contient toutes les caractéristiques RF importantes. Elles peuvent être ajoutées comme cela a été présenté dans le chapitre

précédent. Il convient d'unifier les grandeurs électriques utilisées. Dans ce travail, la NF et l'IIP3 sont notés en dBm, et le gain linéaire en dB, comme le montre la figure 3.2.

Filtrage analogue paramétrable : Dans le récepteur, CXF1 est un filtre passe-bande complexe. Sa fonction de transfert H(s) à coefficients complexes peut être considérée comme un filtre passe-bas avec un décalage en fréquence Δf, où Δf est la fréquence centrale du filtre complexe. Dans le modèle, les caractéristiques analogiques sont ajoutées en début du bloc comme dans le cas du LNA. La deuxième étape est un filtrage « commun » qui prend la fréquence décalée et les coefficients du LPF comme paramètres d'entrée (cf. Fig. 3.3). L'entrée et la sortie étant des signaux (nombres) complexes transmis en quadrature (voies I et Q) par un filtrage complexe, alors le modèle est créé avec les entrées et sorties en quadrature. Dans le cas de la construction d'un filtrage numérique réel, la fréquence décalée et les parties imaginaires de l'entrée et de la sortie doivent être mises en zéro.

Filtrage numérique (ADC et filtres numériques) : Le modèle à construire pour le filtrage numérique doit être commun à tous les blocs pour qu'il puisse être utilisé pour décrire tous les filtrages numériques réels et complexes. Basé sur la méthode de réalisation des filtres numériques décrite dans le paragraphe précédent, ce modèle commun est construit avec des entrées et sorties en quadrature, et les coefficients de filtrage sont des nombres complexes. En considérant la contribution en bruit de l'ADC, ce bloc prend la NF comme paramètre générique optionnel. Nous supposons que le bruit de quantification sera en dehors de la bande d'intérêt en sortie de l'ADC, et que ce bruit total est défini par sa NF. Pour réaliser la mise en forme du bruit, on génère un bruit interne filtré par NTF(z), où δ_q est la densité de bruit de quantification théorique et n'a aucun effet sur le SNR en sortie de l'ADC. Les coefficients du filtre et les signaux en entrée et en sortie sont, là encore, des nombres complexes. Lors de la création d'un ADC ΣΔ, on doit initialiser les valeurs complexes pour générer les coefficients complexes de NTF et de STF. En cas de filtrage numérique, le chemin de NTF doit être mis à zéro en définissant l'ordre du filtre NTF à zéro. En cas de LPF, les paramètres incluant la voie Q sont définis comme des nombres complexes, dont la partie imaginaire est nulle (cf. Fig. 3.4).

Génération de l'ADC ou du CXF2 en utilisant "DigFil": Un exemple de génération automatique d'un ADC de type ΣΔ, dans une netlist d'un récepteur est illustré par le code suivant :

```
SC_MODULE(receiver){
  ……
  SC_CTOR (receiver) {  ……
    adc_sd=new DigFil("ADC",stf_gain,sa,sb,na,nb, 3);
    ……}
  ……}
```

Le spectre de sortie est donné par la figure 3.5. Les résultats de simulation à partir de ce modèle généré permettent par exemple de retrouver la NF donnée en paramètre générique. Ces resultats sont résumés dans le Tableau 3.5. Nous avons également illustré l'utilisation de la fonction "DigFil" pour créer le filtre numérique complexe CXF2, les résultats sont identiques à ceux présentés au chapitre 2. Ceci permet de valider notre fonction de génération automatique de filtres numériques "DigFil".

3. Plateforme SCTLM au niveau LL et interface avec la couche PHY

3.1. Introduction

La plateforme initiale de simulation d'un réseau BLE a été développée par la société Riviera Waves pour tester les applications BLE de ses clients, sans prendre en compte la couche PHY. Ainsi, le niveau d'abstraction le plus bas de cette plateforme est la couche LL. Il est décrit comme une série de "threads" qui définissent les performances BLE de la couche LL selon la spécification de ses états. Dans les threads de LL, on s'intéresse seulement à des opérations où il y a des appels de fonction vers le modèle "canal" qui indiquent l'activation ou la désactivation de l'émetteur-récepteur. On ignore le contrôle ou les erreurs de haut niveau, car ils ne peuvent pas être vus par la couche PHY. Ce modèle original sera conservé dans le nouveau système en incluant l'émetteur-récepteur intégré, mais avec les interfaces de données et de contrôle, nouvellement créées. Il est alors nécessaire de comprendre l'échange de données dans les threads de LL.

3.2. Processus « Passive scan »

Le processus "Passive Scan" est expliqué en détail car il est l'état le plus simple de la couche LL. Il est déclenché par le processus passif "sc_core :: sc_event pscan_evt", alors LL ouvrira une fenêtre

de scan pour attendre le paquet d'advertissing. Ce processus est réalisé en utilisant une boucle qui commence en appelant la fonction rx() dans le modèle "Air" avec la spécification de la longueur de fenêtre de scan "winsize" et qui va attendre jusqu'à la réception d'un paquet ou jusqu'à la fin de la fenêtre de scan. Un exemple est illustré par la figure 3.6

3.3. Air modèle

Le modèle "Air" est le modèle du "canal" de données dans la plateforme BLE originale. Il est composé de deux fonctions principales "tx()" et "rx()" pour la transmission et la réception de données. Pendant la simulation, elles sont appelées à chaque fois qu'il y a des transmissions ou des réceptions dans les threads de LL. Mais actuellement, dans la simulation de haut niveau, les vraies transmissions de données ne sont pas nécessaires. Les échanges de paquets sont décrits comme des événements séquencés. Ce modèle vise aux tests d'erreur de haut niveau, l'échange de données dans ce canal est considéré comme parfait. Un exemple d'une transaction entre deux LL est expliqué pour donner une idée du fonctionnement général du modèle du canal au niveau de la couche LL (cf. Fig. 3.7).

3.4. Création de l'interface entre la couche LL et le transceiver

L'interface de données entre le domaine numérique TLM et le domaine analogique/RF AMS exige une conversion. On appelle cette interface le "chemin de données" (cf. Fig. 3.8). Il convient également de décrire les signaux de contrôle, qui sont délivrés par la couche LL afin de contrôler ou configurer l'émetteur-récepteur RF. Nous l'appellerons le "chemin de contrôle", et il constitue une partie de l'interface.

Chemin de données : Une interface RTL est créée pour réaliser la conversion des données entre les deux domaines. Cette interface convertit le paquet BLE venant du LL en une série de bits afin de les transmettre (dans la chaîne Tx) avec un débit de données de 1 MHz. En réception, les données reçus sont regroupés en paquets. Les paquets sont ensuite transmis vers le LL pour un traitement ultérieur. A la sortie du récepteur (Rx à la fin du démodulateur), les données reçues sont des flux de bits sur-échantillonnés à la fréquence de 13 MHz. Un corrélateur est utilisé à la fin du démodulateur pour détecter les paquets reçus avant de les transmettre au LL (cf. Fig. 3.9). Dans le cas d'une erreur de réception, LL recevra un paquet NULL.

Chemin de contrôle : Il est important de savoir quel est le plus grand consommateur d'énergie dans l'émetteur-récepteur pour une application donnée. La partie RF est en général le principal consommateur d'énergie d'un système de communication. L'estimation de l'énergie de cette partie est basée sur les puissances de chaque bloc RF. Ces valeurs doivent être précises au niveau circuit et elles sont déterminées par la mesure de circuits réels, ou par des simulations au niveau SPICE. Dans ce travail, elles sont fournies par la société Riviera Waves, à partir de mesures sur des prototypes. Pour estimer la consommation d'un système complet lors de simulations réalistes, nous devons prendre en compte l'activité du "Séquenceur", bloc intégré au sein de la puce de l'émetteur-récepteur. Il est utilisé pour activer les blocs RF uniquement lorsque cela est nécessaire, pour éviter un gaspillage de l''énergie au cours de la communication. Le séquenceur modélisé par une machine d'états (module "sc_module") est illustré par la figure 3.10. Les états du séquenceur sont fournis par la couche LL. Les figures 3.10 à 3.13 permettent d'illustrer la mise en place de cette technique de power gating.

3.5. Configuration et intégration du transceiver au sein de la plateforme SCTLM

Le modèle de l'émetteur-récepteur peut être initialisé avec les configurations définies dans le Tableau 3.3, et intégré au sein de la plateforme SCTLM. Le modèle de la chaîne Tx est conservée dans sa version originale, sans modification, car il ne correspond pas aux objectifs de ce travail. Par contre, pour estimer les performances de la chaine de réception, le modèle du Rx est construit à partir des modèles paramétrables présentés au paragraphe 2 de ce chapitre (cf. Fig. 3.14). L'interface pour le connecter à la couche LL contient le chemin de données et le chemin de contrôle. Les valeurs de configuration (paramètres génériques des modèles) sont résumées dans les Tableaux 3.3 et 3.4.

3.6. Simulation d'un système BLE global

Dans ce travail, la simulation globale vise à estimer la consommation en énergie de l'émetteur-récepteur et les performances du Rx. La plateforme est construite simplement à partir de l'architecture d'un réseau avec seulement 2 appareils BLE (DEV0 et DEV1).

Pour réaliser une simulation afin d'estimer la consommation d'énergie, on a besoin tout d'abord d'un cas d'étude ou use case réaliste à partir d'une application BLE ou un scénario personnalisé en respectant la norme de la communication BLE. Les applications BLE sont normalement très courtes dans le temps, on a ainsi choisi de définir un scénario où il y a beaucoup d'échanges de données.

1er use case : Le premier « use case » appliqué à une simulation globale est très court. Il est utilisé pour valider l'estimation de l'énergie consommée. Dans ce use case, DEV1 envoie des paquets de ADV dans les canaux « advertising » 37, 38, 39. DEV0 ouvre une fenêtre de scan pour écouter sur ces canaux d'advertising et recevoir des paquets de ADV envoyés par d'autres dispositifs autour de lui, DEV1 dans cet exemple. À la fin de la fenêtre de scan, DEV0 demande à DEV1 d'établir une connexion. Lorsque cette connexion est réussie, les deux dispositifs sont dans l'état de connexion, où DEV0 joue le rôle de master et DEV1 celui d'esclave. Enfin 100 paquets de données sont envoyés de DEV0 à DEV1.

La simulation est exécutée avec un plancher de bruit de -114 dBm et la puissance du signal d'entrée du Rx varie entre -96 dBm et -91 dBm. Les courbes de puissance « en sortie » de Tx et de Rx sont présentées sur la figure 3.17. Elles montrent que l'estimation de la puissance fonctionne correctement car la courbe change étape par étape sous le contrôle des signaux de contrôle. L'estimation d'énergie est alors obtenue en intégrant les courbes de puissance au cours du temps.

Autres use cases : Le premier use case est trop court pour observer la consommation d'énergie. Il ne sera pas évident d'obtenir la différence de consommation entre les différents émetteurs-récepteurs. D'autres use cases sont utilisés pour une simulation plus réaliste de cas réels. Le 2ième use case présente deux dispositifs qui établissent pendant une courte période une connexion. Puis lorsqu'ils sont connectés, "DEV1" envoie 100 paquets de données à "DEV0".

Le 3ième use case est défini par deux dispositifs qui exécutent plusieurs événements sur un canal d'advertising. Les simulations sont relancées et les résultats sont donnés avec les consommations d'énergie. Les figures 3.18 et 3.19 présentent les résultats de simulation, dont les valeurs principales sont résumés dans les tableaux 3.5 et 3.6.

Nous avons ainsi terminé le travail de modélisation et de vérification pour la simulation globale de systèmes BLE en SC-TLM et SC-AMS. Les performances de l'émetteur-récepteur et l'estimation de la consommation d'énergie au niveau circuit sont obtenues. La simulation est extrêmement plus rapide que celle en utilisant les méthodes de modélisation traditionnelles.

3.7. Optimisation de la consommation d'énergie

Afin de diminuer la consommation en énergie d'un transceiver, la société Rivera Waves a conçu et réalisé un nouveau design de son récepteur. L'objectif de notre simulation est d'anticiper l'amélioration des performances du système par des simulations. Il s'agit donc de vérifier et estimer cette baisse de consommation et voir son impact sur la qualité du bilan de liaison en termes de BER. La nouvelle version optimisée du récepteur correspond aux nouvelles spécifications (caractéristiques) du LNA et du mélangeur dont les valeurs sont résumées dans le tableau 3.7. L'estimation de l'énergie est obtenue avec une seule simulation, par contre l'estimation du BER est réalisée en faisant varier la puissance d'entrée du signal pour calculer la sensibilité optimale offrant un BER de 0,1%. Les résultats de simulation sont présentés sur la figure 3.21 et résumés dans les tableaux 3.8 et 3.9.

Ces résultats montre que le récepteur optimisé consomme 16% d'énergie en moins que l'ancienne version sans dégrader la qualité de liaison (BER). Au contraire, sa sensibilité est améliorée de 2dB.

4. Modélisation et simulation d'un WSN

Jusqu'à présent, les cas d'études se sont restreints à une communication point à point entre deux dispositifs BLE. L'objectif de ce dernier paragraphe, qui constitue également une piste pour des travaux futurs (i.e. des perspectives), est de réaliser une dernière simulation à un niveau d'abstraction encore plus élevée, à savoir un réseau de capteurs (ou wireless sensor network, WSN). La figure 3.22 illustre un exemple d'un tel cas d'étude.

En collaboration avec l'Université de Vérone, nous avons interfacé notre plateforme SCTLM/SCAMS de simulation BLE avec la plateforme SCNSL, développée par nos collègues italiens. Noter approche modulaire et la méthodologie MIM nous ont permis de réaliser cet interfaçage sans réelle difficulté. Les résultats obtenus sont présentés dans le tableau 3.10.

5. Conclusion

Dans ce chapitre, les modèles configurables de la chaîne de réception RF ont été développés. Ensuite, l'émetteur-récepteur basé sur ces modèles est construit et intégré dans un système BLE décrit en SC-TLM. Pour être capable de connecter le domaine SC-AMS au domaine SC-TLM, nous avons créé des interfaces qui comportent le chemin de données et le chemin de contrôle. Nous avons ainsi pu réaliser une simulation globale pour un réseau BLE de deux dispositifs. Elle a été lancée avec différents « use cases », afin de déterminer les consommations d'énergie suivant différentes applications. En utilisant les mêmes cas d'études, nous avons comparé deux versions d'une même architecture de récepteurs et avons pu montrer par simulation le comportement énergétique de chacun, ainsi que leur sensibilité, répondant ainsi aux demandes de notre partenaire industriel. En effet, cette plateforme de simulation permet d'anticiper le comportement d'un circuit ou d'une architecture pour une application réaliste afin de pouvoir optimiser une chaine complète et non des composants séparément où une optimisation « locale » (au niveau circuit) peut s'avérer inefficace au niveau système.

Enfin, nous avons proposé une piste d'évolution de cette plateforme en simulant non plus une communication point à point entre deux dispositifs BLE, mais en simulant un réseau de capteurs. Ainsi, en collaboration avec l'Université de Vérone, nous avons interfacé notre plateforme SCTLM/SCAMS de simulation BLE avec la plateforme SCNSL, développée par nos collègues italiens.

Conclusions et travaux futurs

1. Conclusion générale

Nous avons présenté, dans ce travail de thèse, une approche basée sur la méthodologie de développement dite Meet-in-the-Middle (MIM) pour simuler rapidement et efficacement des systèmes de communication sans fil. Ce type de simulation est basé sur une modélisation au niveau système, tout en utilisant des niveaux d'abstraction différents par extraction de caractéristiques de niveau inférieur dans le but d'optimiser les performances du système. Ce travail a consisté à modéliser un transceiver RF BLE et à en analyser les performances en utilisant une plateforme écrite en SC-TLM, en prenant en compte certaines spécifications de la couche PHY, décrite en SC-AMS.

Nous avons rappelé les méthodologies concernant les approches de modélisation et les considérations relatives aux circuits et systèmes RF. Dans le premier chapitre, l'approche MIM combinée avec la modélisation équivalente en bande de base (BBE) a été choisie parmi les méthodes de modélisation souvent utilisées pour les systèmes mixtes, pour réaliser une modélisation du transceiver RF au niveau de système. Ce modèle vise à une simulation avec une précision de bas niveau et une vitesse de simulation du niveau système. Cette méthode nécessite de prendre en compte uniquement les spécifications nécessaires aux objectifs fixés (par exemple détermination du couple consommation, BER) à partir des conceptions RF de bas niveau. Ces spécifications RF concernant le gain linéaire, les non-linéarités et le bruit intrinsèque sont introduites. Ensuite, le standard BLE est décrit succinctement. La couche de Link Layer est présentée plus en détail puisqu'elle sert d'interface entre les couches supérieures (numériques et décrites en SC-TLM) et la couche physique (principalement analogique et RF et qui sera décrite en SC-AMS). A la fin de ce chapitre, les outils et les langages souvent utilisés pour la modélisation de signaux mixtes sont également présentés. Finalement, SC-AMS est choisi parmi eux comme l'outil de modélisation pour construire notre modèle de transceiver, en raison de son efficacité et son interopérabilité avec les outils SystemC.

Le second chapitre vise à modéliser un transceiver BLE complet en SC-AMS. Un premier modèle fonctionnel a été développé, puis raffiné en prenant en compte les principales spécifications RF mesurées à partir des vrais circuits, réalisés par la société Riviera Waves. Une optimisation de la vitesse de simulation (par un facteur de 80) a été effectuée, en utilisant l'approche BBE, afin de pouvoir réaliser ultérieurement des simulations réalistes de cas

d'études fournis par notre partenaire industriel. Ainsi, nous avons pu déterminer la relation entre le BER du transceiver et la puissance du signal d'entrée du récepteur ou son SNR.

Dans le chapitre 3, nous avons d'abord développé les modèles configurables pour les blocs RF de la chaîne Rx. Ensuite, le modèle du LL et le modèle de l'Air d'origine en SC-TLM sont présentés en détail pour préparer la création de l'interface LL/PHY, contenant les chemins de données et de contrôle. La plateforme BLE a été ensuite complétée en connectant le domaine SC-TLM avec notre modèle de transceiver en SC-AMS. Des cas d'études d'applications BLE ont été réalisés et ont permis de déterminer les performances en termes de consommation d'énergie des parties Rx et Tx du transceiver et de bilan de liaison (BER). Nous avons ainsi pu comparer également deux versions d'une même architecture de récepteur où les caractéristiques du LNA et du mélangeur sont différentes. Les simulations ont permis de valider les gains en performance au niveau système (consommation réduite de 16% et amélioration de la sensibilité de 2dB) et donc une optimisation globale par une modification au niveau circuit. Ces résultats ont été validés par les mesures sur les circuits développés par la société Riviera Waves. Pour conclure ce chapitre, nous avons proposé une première piste de poursuite de ces travaux, à savoir interfacer notre plateforme SC-TLM/SC-AMS avec un outil de simulation de réseau. Nous avons montré qu'il était possible de réutiliser nos modèles pour une simulation très haut niveau (réseau de capteurs) en les intégrant dans la plateforme SCNSL, développée par l'université de Vérone, avec qui nous avons collaboré.

2. Perspectives

Dans le travail de modélisation, la PLL a été représentée par un bruit de phase qui est décrit par un modèle théorique. Pour obtenir un bruit de phase plus précis, la PLL peut être modélisée comme un système complet avec les détails de tous les sous-blocs de la PLL.

Les interfaces dans ce travail ont été faites uniquement pour notre système BT ou BLE. Pour obtenir une interface plus commune, nous pouvons modifier la partie LL pour la connecter avec un transceiver plus facilement et ainsi viser d'autres standards.

L'ensemble des modèles configurables peut être complété avec d'autres développements possibles pour satisfaire les exigences d'autres modèles d'architecture. Les méthodes de modélisation présentées dans ce travail peuvent être mises en œuvre pour d'autres projets, afin de modéliser les systèmes d'autres standards de communication par exemple ou d'autres applications (cas d'études plus complexes comme les réseaux de capteurs). Une première intégration dans un réseau de capteurs sans fil a été réalisée et présenté avec la plateforme SCNSL. Il conviendrait d'étendre cet interfaçage à d'autres simulateurs réseaux comme OMNeT ++ ou WSNet..

Simulation multi-moteurs multi-niveaux pour la validation des spécifications système et optimisation de la consommation

Ce travail vise la modélisation au niveau système, en langage SystemC-AMS, et la simulation d'un émetteur-récepteur au standard Bluetooth Low Energy (BLE). L'objectif est d'analyser la relation entre les performances, en termes de BER et la consommation d'énergie du transceiver. Le temps de simulation d'un tel système, à partir de cas d'étude (use case) réaliste, est un facteur clé pour le développement d'une telle plateforme. De plus, afin d'obtenir des résultats de simulation le plus précis possible, les modèles « haut niveau » doivent être raffinés à partir de modèles plus bas niveau où de mesure. L'approche dite Meet-in-the-Middle, associée à la méthode de modélisation équivalente en Bande Base (BBE, BaseBand Equivalent), a été choisie pour atteindre les deux conditions requises, à savoir temps de simulation « faible » et précision des résultats. Une simulation globale d'un système de BLE est obtenue en intégrant le modèle de l'émetteur-récepteur dans une plateforme existante développée en SystemC-TLM. La simulation est basée sur un système de communication de deux dispositifs BLE, en utilisant différents scénarios (différents cas d'utilisation de BLE).

Dans un premier temps nous avons modélisé et validé chaque bloc d'un transceiver BT. Devant le temps de simulation prohibitif, les blocs RF sont réécrits en utilisant la méthodologie BB, puis raffinés afin de prendre en compte les non-linéarités qui vont impacter le couple consommation, BER. Chaque circuit (chaque modèle) est vérifié séparément, puis une première simulation système (point à point entre un émetteur et un récepteur) est effectuée. Enfin, le BER est estimé. L'ensemble est fonctionnel, les temps de simulation acceptable et les résultats ont été validés à partir de mesure sur un circuit de la société Riviera Waves. Finalement, deux versions d'une même architecture sont modélisées, simulées et comparées. Les résultats montrent que la nouvelle version permet d'augmenter les performances globales du système, à la fois en terme de BER et de consommation. Pour conclure ce travail, nous avons montré qu'il était possible de réutiliser nos modèles pour une simulation très haut niveau (réseau de capteurs) en les intégrant dans la plateforme SCNSL, développée par l'université de Vérone, avec qui nous avons collaboré.

Multi-engine multi-level simulation for system specification validation and power consumption optimization

This work aims at system-level modelling a defined transceiver for Bluetooth Low energy (BLE) system using SystemC-AMS. The goal is to analyze the relationship between the transceiver performance and the accurate energy consumption. This requires the transceiver model contains system-level simulation speed and the low-level design block power consumption and other RF specifications. The Meet-in-the-Middle approach and the Baseband Equivalent method are chosen to achieve the two requirements above. A global simulation of a complete BLE system is achieved by integrating the transceiver model into a SystemC-TLM described BLE system model which contains the higher-than-PHY levels. The simulation is based on a two BLE devices communication system and is run with different BLE use cases. The transceiver Bit-Error-Rate and the energy estimation are obtained at the end of the simulation.

First, we modelled and validated each block of a BT transceiver. In front of the prohibitive simulation time, the RF blocks are rewritten by using the BBE methodology, and then refined in order to take into account the non-linearities, which are going to impact the couple consumption, BER. Each circuit (each model) is separately verified, and then a first BLE system simulation (point-to-point between a transmitter and a receiver) has been executed. Finally, the BER is finally estimated. This platform fulfills our expectations, the simulation time is suitable and the results have been validated with the circuit measurement offered by Riviera Waves Company. Finally, two versions of the same transceiver architecture are modelled, simulated and compared. The results show that the new version improves the global performance (both in BER and power consumption. To conclude the work, we have shown that it was possible to reuse our models for a very high level (network of sensors) simulation by integrating them into a platform of SCNSL which is developed by University of Verona (Italy), with whom we collaborated.